# Background Statement for SEMI Draft Document 5549A

# REVISION TO SEMI E30-0416
# GENERIC MODEL FOR COMMUNICATIONS AND CONTROL OF MANUFACTURING EQUIPMENT (GEM) with Title Change to SPECIFICATION FOR THE GENERIC MODEL FOR COMMUNICATIONS AND CONTROL OF MANUFACTURING EQUIPMENT (GEM)

**NOTICE:** This Background Statement is not part of the balloted item. It is provided solely to assist the recipient in reaching an informed decision based on the rationale of the activity that preceded the creation of this ballot.

**NOTICE:** For each Reject Vote, the Voter shall provide text or other supportive material indicating the reason(s) for disapproval (i.e., Negative[s]), referenced to the applicable section(s) and/or paragraph(s), to accompany the vote.

**NOTICE:** Recipients of this ballot are invited to submit, with their Comments, notification of any relevant patented technology or copyrighted items of which they are aware and to provide supporting documentation. In this context, 'patented technology' is defined as technology for which a patent has been issued or has been applied for. In the latter case, only publicly available information on the contents of the patent application is to be provided.

**Background**

The introductory sections of SEMI E30 are currently not arranged in the same order as most published SEMI Standards and Safety Guidelines (e.g., § 1, Purpose; § 2, Scope). Rearranging these sections for consistency with other published SEMI Standards would not alter the technical content, but it would affect the numbering of most section headings.

In this ballot:

- Section 4 and after are essentially unchanged
    - References to the Related Information are updated
    - There is a spelling correction (Requirements)
- Sections 1 to 3 are mostly just a rearrangement of unchanged text
    - The "Purpose" section consists of Sections ¶ 1.3 – 1.3.5 from E30-0416.
    - "Scope" is now a Level 1 Heading, and uses ¶ 1.2. from E30-0416.
    - The new "Limitations" section consists of ¶ 1.2.1 and 1.2.2 from E30-0416.
    - "Overview" has been deleted as its essential content is covered by the Table of Contents. This deletion is the only content change.
    - "Application Notes" has been changed to "Related Information". References updated throughout document.
    - 'EnableSpooling" EC has been added to the Variable Item List in section 8.4.2 under the list of ECVs. This is an EC required by Spooling, but previously had failed to be listed.

Table of Contents and internal cross reference numbering will be updated after ballot is approved for publication. Please ignore any mistake right now in cross references.

E30 is referenced by other Standards, not only those from I&CC, but from other committees as well (e.g., PV). Some of these references actually call out specific sections of E30.

The ballot also proposes a title change by adding the appropriate sub-type (i.e., Specification) per the SEMI Standards Procedure Guide (Appendix A2-1).

Note that text to be added are shown in underlined blue text. Deleted items are shown in ~~strikethrough red text~~.

**Revision Control**

**This revision control records activity within the task force as well as formal submit and resubmit dates and results per SEMI. Entries have been made by the task force.**

| Date | Version | Name | Edits |
|---|---|---|---|
| Jan 11, 2017 | 1.0 | Brian Rubow | Initial edits to adapt the original 5549 changes to the 0416 E30 specification. |

The ballot results will be reviewed and adjudicated at the meetings indicated in the table below. Check www.semi.org/standards under Standards Calendar for the latest update.

### Review and Adjudication Information

| | Task Force Review | Committee Adjudication |
|---|---|---|
| **Group:** | NA GEM300 TF | NA Information & Control Committee |
| **Date:** | April 4th 2017 | April 5th 2017 |
| **Time & Timezone:** | 13:00 – 16:00 Pacific Time | 8:00 – 17:00 Pacific Time |
| **Location:** | SEMI Headquarters | SEMI Headquarters |
| **City, State/Country:** | Milpitas, CA/USA | Milpitas, CA/USA |
| **Leader(s):** | Brian Rubow (Cimetrix) | Jack Ghiselli (Ghiselli Consulting)<br>Brian Rubow (Cimetrix)<br>James Moyne (University of Michigan) |
| **Standards Staff:** | Inna Skvortsova (SEMI NA)<br>1.408.943.6996<br>iskvortsova@semi.org | Inna Skvortsova (SEMI NA)<br>1.408.943.6996<br>iskvortsova@semi.org |

This meeting's details are subject to change, and additional review sessions may be scheduled if necessary. Contact the task force leaders or Standards staff for confirmation.

Telephone and web information will be distributed to interested parties as the meeting date approaches. If you will not be able to attend these meetings in person but would like to participate by telephone/web, please contact Standards staff.

SEMI
673 S. Milpitas Blvd.
Milpitas, CA 95035-5446
Phone: 408.943.69003

DRAFT
Document Number: 5549A
Date: 2/15/2017

LETTER BALLOT

# SEMI Draft Document 5549A

## REVISION TO SEMI E30-0416 GENERIC MODEL FOR COMMUNICATIONS AND CONTROL OF MANUFACTURING EQUIPMENT (GEM) with title change to

## SEMI E30-0416: SPECIFICATION FOR THE GENERIC MODEL FOR COMMUNICATIONS AND CONTROL OF MANUFACTURING EQUIPMENT (GEM)

**NOTICE: Additions are indicated by underline and deletions are indicated by ~~strikethrough~~.**

This Standard was technically approved by the Information & Control Global Technical Committee. This edition was approved for publication by the global Audits and Reviews Subcommittee on December 20, 2012. Available at www.semiviews.org and www.semi.org in April 2016; originally published in 1992; previously published June 2011.

### Table of Contents

LETTER BALLOT

## 1 Purpose ~~Introduction~~

1.1 ~~*Revision History* This is the first release of the GEM Standard.~~ GEM defines a standard implementation of SECS-II for all semiconductor manufacturing equipment. The GEM Standard defines a common set of equipment behavior and communications capabilities that provide the functionality and flexibility to support the manufacturing automation programs of semiconductor device manufacturers. Equipment suppliers may provide additional SECS-II functionality not included in GEM as long as the additional functionality does not conflict with any of the behavior or capabilities defined in GEM. Such additions may include SECS-II messages, collection events, alarms, remote command codes, processing states, variable data items (data values, status values or equipment constants), or other functionality that is unique to a class (etchers, steppers, etc.) or specific instance of equipment.

1.1.1 GEM is intended to produce economic benefits for both device manufacturers and equipment suppliers. Equipment suppliers benefit from the ability to develop and market a single SECS-II interface that satisfies most customers. Device manufacturers benefit from the increased functionality and standardization of the SECS-II interface across all manufacturing equipment. This standardization reduces the cost of software development for both equipment suppliers and device manufacturers. By reducing costs and increasing functionality, device manufacturers can automate semiconductor factories more quickly and effectively. The flexibility provided by the GEM Standard also enables device manufacturers to implement unique automation solutions within a common industry framework.

1.1.2 The GEM Standard is intended to specify the following:

- A model of the behavior to be exhibited by semiconductor manufacturing equipment in a SECS-II communication environment,
- A description of information and control functions needed in a semiconductor manufacturing environment,
- A definition of the basic SECS-II communications capabilities of semiconductor manufacturing equipment,
- A single consistent means of accomplishing an action when SECS-II provides multiple possible methods, and
- Standard message dialogues necessary to achieve useful communications capabilities.

1.1.3 The GEM Standard contains two types of requirements:

- fundamental GEM requirements and
- requirements of additional GEM capabilities.

1.1.4 The fundamental GEM requirements form the foundation of the GEM Standard. The additional GEM capabilities provide functionality required for some types of factory automation or functionality applicable to specific types of equipment. A detailed list of the fundamental GEM requirements and additional GEM capabilities can be found in § 11, GEM Components ~~Compliance~~.Figure 1 illustrates the components of the GEM Standard.

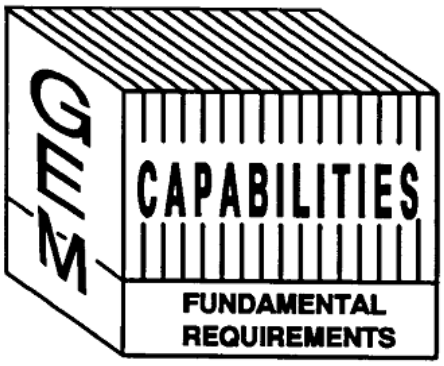



**Figure 1**
**GEM Components**

LETTER BALLOT

1.1.5 Equipment suppliers should work with their customers to determine which additional GEM capabilities should be implemented for a specific type of equipment. Because the capabilities defined in the GEM Standard were specifically developed to meet the factory automation requirements of semiconductor manufacturers, it is anticipated that most device manufacturers will require most of the GEM capabilities that apply to a particular type of equipment. Some device manufacturers may not require all the GEM capabilities due to differences in their factory automation strategies.

## 2 ~~*1.2*~~ *Scope*

2.1 The scope of the GEM Standard is limited to defining the behavior of semiconductor equipment as viewed through a communications link. The SEMI E5 (SECS-II) Standard provides the definition of messages and related data items exchanged between host and equipment. The GEM Standard defines which SECS-II messages should be used, in what situations, and what the resulting activity should be. Figure 1 illustrates the relationship of GEM, SECS-II and other communications alternatives.

~~2.1.1 1.2.1 The GEM Standard does NOT attempt to define the behavior of the host computer in the communications link. The host computer may initiate any GEM message scenario at any time and the equipment shall respond as described in the GEM Standard. When a GEM message scenario is initiated by either the host or equipment, the equipment shall behave in the manner described in the GEM Standard when the host uses the appropriate GEM messages.~~

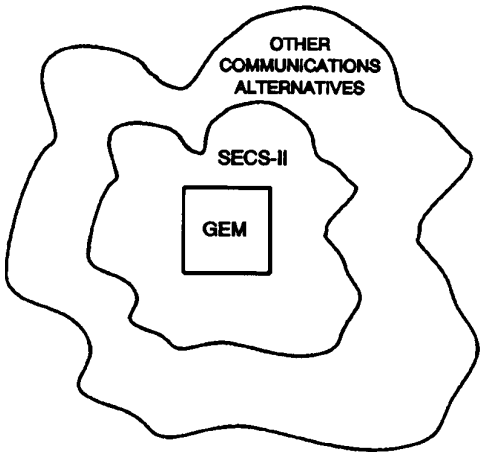



**Figure 2**
**GEM Scope**

## 3 *Limitations*

3.1 The GEM Standard does NOT attempt to define the behavior of the host computer in the communications link. The host computer may initiate any GEM message scenario at any time and the equipment shall respond as described in the GEM Standard. When a GEM message scenario is initiated by either the host or equipment, the equipment shall behave in the manner described in the GEM Standard when the host uses the appropriate GEM messages.

3.2 ~~1.2.2~~ The capabilities described in this Standard are specifically designed to be independent of lower-level communications protocols and connection schemes (e.g., SECS-I, SMS, point-to-point, connection-oriented or connectionless). Use of those types of standards is not required or precluded by this Standard.

**NOTICE:** SEMI Standards and Safety Guidelines do not purport to address all safety issues associated with their use. It is the responsibility of the users of the Documents to establish appropriate safety and health practices, and determine the applicability of regulatory or other limitations prior to use.

~~*1.3 Intent* — GEM defines a standard implementation of SECS-II for all semiconductor manufacturing equipment. The GEM Standard defines a common set of equipment behavior and communications capabilities that provide the~~

LETTER BALLOT

~~functionality and flexibility to support the manufacturing automation programs of semiconductor device manufacturers. Equipment suppliers may provide additional SECS-II functionality not included in GEM as long as the additional functionality does not conflict with any of the behavior or capabilities defined in GEM. Such additions may include SECS-II messages, collection events, alarms, remote command codes, processing states, variable data items (data values, status values or equipment constants), or other functionality that is unique to a class (etchers, steppers, etc.) or specific instance of equipment.~~

~~1.3.1 GEM is intended to produce economic benefits for both device manufacturers and equipment suppliers. Equipment suppliers benefit from the ability to develop and market a single SECS-II interface that satisfies most customers. Device manufacturers benefit from the increased functionality and standardization of the SECS-II interface across all manufacturing equipment. This standardization reduces the cost of software development for both equipment suppliers and device manufacturers. By reducing costs and increasing functionality, device manufacturers can automate semiconductor factories more quickly and effectively. The flexibility provided by the GEM Standard also enables device manufacturers to implement unique automation solutions within a common industry framework.~~

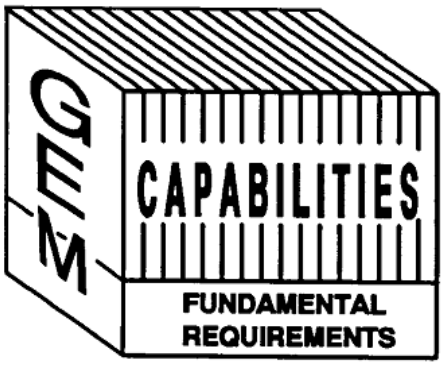
~~1.3.2 The GEM Standard is intended to specify the following:~~

- ~~A model of the behavior to be exhibited by semiconductor manufacturing equipment in a SECS-II communication environment,~~
- ~~A description of information and control functions needed in a semiconductor manufacturing environment,~~
- ~~A definition of the basic SECS-II communications capabilities of semiconductor manufacturing equipment,~~
- ~~A single consistent means of accomplishing an action when SECS-II provides multiple possible methods, and~~
- ~~Standard message dialogues necessary to achieve useful communications capabilities.~~

~~1.3.3 The GEM Standard contains two types of requirements:~~

- ~~fundamental GEM requirements and~~
- ~~requirements of additional GEM capabilities.~~

~~1.3.4 The fundamental GEM requirements form the foundation of the GEM Standard. The additional GEM capabilities provide functionality required for some types of factory automation or functionality applicable to specific types of equipment. A detailed list of the fundamental GEM requirements and additional GEM capabilities can be found in § 9, GEM Compliance. Figure 2 illustrates the components of the GEM Standard.~~




**~~Figure 3~~**
**~~GEM Components~~**

~~1.3.5 Equipment suppliers should work with their customers to determine which additional GEM capabilities should be implemented for a specific type of equipment. Because the capabilities defined in the GEM Standard were specifically developed to meet the factory automation requirements of semiconductor manufacturers, it is anticipated that most device manufacturers will require most of the GEM capabilities that apply to a particular type of equipment.~~

LETTER BALLOT

~~Some device manufacturers may not require all the GEM capabilities due to differences in their factory automation strategies.~~

~~*1.4 Overview* — The GEM Standard is divided into sections as described below.~~

~~*1.4.1 § 1 — Introduction*~~

~~1.4.1.1 This section provides the revision history, scope and intent of the GEM Standard. It also provides an overview of the structure of the Document and a list of related documents.~~

~~*1.4.2 § 2 — Definitions*~~

~~*1.4.2.1* This section provides definitions of terms used throughout the Document.~~

~~*1.4.3 § 3 — State Models*~~

~~1This section describes the conventions used throughout this Document to depict state models. It also describes the basic state models that apply to all semiconductor manufacturing equipment and that pertain to more than a single capability. State models describe the behavior of the equipment from a host perspective.~~

~~*1.4.4 § 4 — Capabilities and Scenarios*~~

~~1.4.4.1This section provides a detailed description of the communications capabilities defined for semiconductor manufacturing equipment. The description of each capability includes the purpose, definitions, requirements, and scenarios that shall be supported.~~

~~*1.4.5 § 5 — Data Definitions*~~

~~1.4.5.1 This section provides a reference to the Data Item Dictionary and Variable Item Dictionary found in SEMI E5. The first subsection shows those data items from SECS-II which have been restricted in their use (i.e., allowed formats). The second subsection lists variable data items that are available to the host for data collection and shows any restrictions on their SECS-II definitions.~~

~~*1.4.6 § 6 — Collection Events*~~

~~This section provides a list of required collection events and their associated data.~~

~~*1.4.7 § 7 — SECS Message Subset*~~

~~1.4.7.1 This section provides a composite list of the SECS-II messages required to implement all capabilities defined in the GEM Standard.~~

~~*1.4.8 § 8 — GEM Compliance*~~

~~1.4.8.1 This section describes the fundamental GEM requirements and additional GEM capabilities and provides references to other sections of the Standard where detailed requirements are located. This section also defines standard terminology and documentation that can be used by equipment suppliers and device manufacturers to describe compliance with this Standard.~~

~~*1.4.9 § A — Application Notes*~~

~~1.4.9.1 These sections provide additional explanatory information and examples.~~

~~*1.4.10 § A.1 — Factory Operational Script*~~

~~3.2.1.1 This section provides an overview of how the required SECS capabilities may be used in the context of a typical factory operation sequence. This section is organized according to the sequence in which actions are typically performed.~~

~~*1.4.11 § A.2 — Equipment Front Panel*~~

~~1.4.11.1 This section provides guidance in implementing the required front panel buttons, indicators, and switches as defined in this Document. A summary of the front panel requirements is provided.~~

~~*1.4.12 § A.3 — Examples of Equipment Alarms*~~

~~1.4.12.1 This section provides examples of alarms related to various equipment configurations.~~

LETTER BALLOT

~~*1.4.13 § A.4 — Trace Data Collection Example*~~

~~1.4.13.1 This section provides an example of trace initialization by the host and the periodic trace data messages that might be sent by the equipment.~~

~~*1.4.14 § A.5 — Harel Notation*~~

~~1.4.14.1 This section explains David Harel's 'Statechart' notation that is used throughout this Document to depict state models.~~

~~*1.4.15 § A.6 — Example Control Model Application*~~

~~1.4.15.1 This section provides one example of a host's interaction with an equipment's control model.~~

~~*1.4.16 § A.7 — Examples of Limits Monitoring*~~

~~1.4.16.1 This section contains four limits monitoring examples to help clarify the use of limits and to illustrate typical applications.~~

## 4 Referenced Standards and Documents

4.1 *SEMI Standards and Safety Guidelines*

4.1.1 The following SEMI Standards are related to the GEM Standard. The specific portions of these Standards referenced by GEM constitute provisions of the GEM Standard.

SEMI E4 — SEMI Equipment Communications Standard 1 Message Transfer (SECS-I)

SEMI E5 — SEMI Equipment Communications Standard 2 Message Content (SECS-II)

SEMI E23 — Specification for Cassette Transfer Parallel I/O Interface

SEMI E37 — High-Speed SECS Message Services (HSMS) Generic Services

SEMI E139 — Specification for Recipe and Parameter Management (RaP)

4.1.2 *Other Documents*

Harel, D. "*Statecharts: A Visual Formalism for Complex Systems*." Science of Computer Programming 8 (1987): pp. 231–274.[1]

**NOTICE:** Unless otherwise indicated, all documents cited shall be the latest published versions.

## 5 Terminology

5.1 *Definitions*

5.1.1 *alarm* — an alarm is related to any abnormal situation on the equipment that may endanger people, equipment, or material being processed. Such abnormal situations are defined by the equipment manufacturer based on physical safety limitations. Equipment activities potentially impacted by the presence of an alarm shall be inhibited.

5.1.1.1 *Discussion* — Note that exceeding control limits associated with process tolerance does not constitute an alarm nor do normal equipment events such as the start or completion of processing.

5.1.2 *capabilities* — capabilities are operations performed by semiconductor manufacturing equipment. These operations are initiated through the communications interface using sequences of SECS-II messages (or scenarios). An example of a capability is the setting and clearing of alarms.

5.1.3 *collection event* — a collection event is an event (or grouping of related events) on the equipment that is considered to be significant to the host.

5.1.4 *communication failure* — a communication failure is said to occur when an established communications link is broken. Such failures are protocol specific. Refer to the appropriate protocol Standard (e.g., SEMI E4 or SEMI E37) for a protocol-specific definition of communication failure.

---

[1] ~~Elsevier Science, P.O. Box 945, New York, NY 10159-0945, USA. http://www.elvesier.nl~~

5.1.5 *communication fault* — a communication fault occurs when the equipment does not receive an expected message, or when either a transaction timer or a conversation timer expires.

5.1.6 *control* — to control is to exercise directing influence.

5.1.7 *equipment model* — an equipment model is a definition based on capabilities, scenarios, and SECS-II messages that manufacturing equipment should perform to support an automated manufacturing environment (see also Generic Equipment Model.).

5.1.8 *event* — an event is a detectable occurrence significant to the equipment.

5.1.9 *GEM Compliance* — the term 'GEM Compliance' is defined with respect to individual GEM capabilities to indicate adherence to the GEM Standard for a specific capability. § 9 includes more detail on GEM Compliance.

5.1.10 *Generic Equipment Model* — the Generic Equipment Model is used as a reference model for any type of equipment. It contains functionality that can apply to most equipment, but does not address unique requirements of specific equipment.

5.1.11 *host* — SEMI E4 and SEMI E5 define Host as "the intelligent system that communicates with the equipment."

5.1.12 *message fault* — a message fault occurs when the equipment receives a message that it cannot process because of a defect in the message.

5.1.13 *operational script* — an operational script is a collection of scenarios arranged in a sequence typical of actual factory operations. Example sequences are system initialization power up, machine setup, and processing.

5.1.14 *operator* — a human who operates the equipment to perform its intended function (e.g., processing). The operator typically interacts with the equipment via the equipment supplied operator console.

5.1.15 *process unit* — a process unit refers to the material that is typically processed as a unit via single run command, process program, etc. Common process units are wafers, cassettes, magazines, and boats.

5.1.16 *processing cycle* — a processing cycle is a sequence wherein all of the material contained in a typical process unit is processed. This is often used as a measure of action or time.

5.1.17 *scenario* — a scenario is a group of SECS-II messages arranged in a sequence to perform a capability. Other information may also be included in a scenario for clarity.

5.1.18 *SEMI Equipment Communications Standard 1 (SECS-I), SEMI E4* — this Standard specifies a method for a message transfer protocol with electrical signal levels based upon EIA RS232-C.

5.1.19 *SEMI Equipment Communications Standard 2 (SECS-II), SEMI E5* — this Standard specifies a group of messages and the respective syntax and semantics for those messages relating to semiconductor manufacturing equipment control.

5.1.20 *SECS Message Service (SMS)* — an alternative to SECS-I to be used when sending SECS-II formatted messages over a network.

5.1.21 *state model* — a state model is a collection of states and state transitions that combine to describe the behavior of a system. This model includes definition of the conditions that delineate a state, the actions/reactions possible within a state, the events that trigger transitions to other states, and the process of transitioning between states.

5.1.22 *system default* — refers to state(s) in the equipment behavioral model that are expected to be active at the end of system initialization. It also refers to the value(s) that specified equipment variables are expected to contain at the end of system initialization.

5.1.23 *system initialization* — the process that an equipment performs at power-up, system activation, and/or system reset. This process is expected to prepare the equipment to operate properly and according to the equipment behavioral models.

5.1.24 *user* — a human or humans who represent the factory and enforce the factory operation model. A user is considered to be responsible for many setup and configuration activities that cause the equipment to best conform to factory operations practices.

## 6 State Models

6.1 The following sections contain state models for semiconductor manufacturing equipment. These state models describe the behavior of the equipment from a host perspective in a compact and easy to understand format. State models for different equipment will be identical in some areas (e.g., communications), but may vary in other areas (e.g., processing). It is desirable to divide the equipment into parallel components that can be modeled separately and then combined. An example of a component overview of an equipment is provided as Figure 3.

6.2 Equipment manufacturers must document the operational behavior of their equipment using state model methodology. State models are discussed in § 4.3 and § A.5, and in a referenced article. Documentation of a state model shall include the following three elements:

- A *state diagram* showing the possible states of the system or components of a system and all of the possible transitions from one state to another. The states and transitions must each be labeled. Use of the Harel notation (see § A.5) is recommended.
- A *transition table* listing each transition, the beginning and end states, what stimulus triggers the transition, and any actions taken as a result of the transition.
- A *definition of each state* specifying system behavior when that state is active.

6.2.1 Examples of the above elements are provided in § A.5.

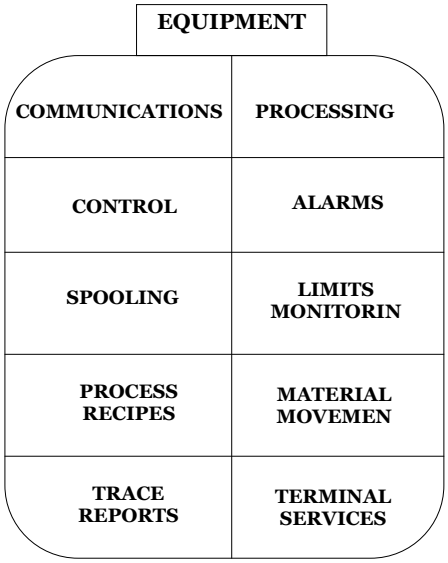



**Figure 4**
**Example Equipment Component Overview**

6.2.2 The benefits of providing state models are:

1. State machine models are a useful specification tool,
2. A host system can anticipate machine behavior based upon the state model,
3. End-users and equipment programmers have a common description of machine behavior from which to work,
4. ‘Legal’ operations can be defined pertaining to any machine state,
5. External event notifications can be related to internal state transitions,
6. External commands can be related to state transitions,

7. State model components describing different aspects of machine control can be related to one another (example: processing state model with material transport state model; processing state model with internal machine safety systems).

6.3 *State Model Methodology* — To document the expected functionality of the various capabilities described in this Document, the 'Statechart' notation developed by David Harel has been adopted. An article by Harel is listed in § 2 and should be considered 'must' reading for a full understanding of the notation. The convention used in this and following sections is to describe the dynamic functionality of a capability with three items: a textual description of each state or substate defined, a table that describes the possible transitions from one state to another, and a graphical figure that uses the symbols defined by Harel to illustrate the relationships of the states and transitions. The combination of these items define the state model for a system or component. A summary of the Harel notation and a more detailed description of the text, table, and figure used to define behavior with this methodology is contained in the Related Information § R1-5 ~~Application Note § A.5~~.

6.3.1 The basic unit of a state model is the state. A state is a static set of conditions. If the conditions are met, the state is current. These conditions might involve sensor readings, switch positions, time of day, etc. Also part of a state definition is a description of reactions to specific stimuli (e.g., if message Sx,Fy is received, generate reply message Sx,Fy + 1). Stimuli may be quite varied but for semiconductor equipment would include received SECS messages, expired timers, operator input at an equipment terminal, and changes in sensor readings.

6.3.2 To help clarify the interpretation of this Document and the state models described herein, it is useful to distinguish between a state and an event and the relationship of one to the other. An event is dynamic rather than static. It represents a change in conditions, or more specifically, the awareness of such a change. An event might involve a sensor reading exceeding a limit, a switch changing position, or a time limit exceeded.

6.3.3 A change to a new active state (state transition) must always be prompted by a change in conditions, and thus an event. In addition, a state transition may itself be termed an event. In fact, there are many events that may occur on an equipment, so it is important to classify events based on whether they can be detected and whether they are of interest. In this Document, the term event has been more narrowly defined as a detectable occurrence that is significant to the equipment.

6.3.4 A further narrowing of the definition of event is represented by the term 'collection event', which is an event (or group of related events) on the equipment that is considered significant to the host. It is these events that (if enabled) are reported to the host. By this definition, the list of collection events for an equipment would typically be only a subset of total events. The state models in this Document are intended to be limited to the level of detail in which the host is interested. Thus, all state transitions defined in this Standard, unless otherwise specified, shall correspond to collection events.

6.4 *Communications State Model* — The Communications State Model defines the behavior of the equipment in relation to the existence or absence of a communications link with the host. § 5.2 expands on this section by defining the Establish Communications capability. This model pertains to a logical connection between equipment and host rather than a physical connection.

6.4.1 *Terminology* — The terms communication failure, connection transaction failure, and communication link are defined for use within this Document only and should not be confused with the same or similar terms used elsewhere.

- See SEMI E4 (SECS-I) or SEMI E37 (HSMS) for a protocol specific definitions of communications failure.
- A connection transaction failure occurs when attempting to establish communications and is caused by:
  - a communication failure,
  - the failure to receive an S1,F14 reply within a reply timeout limit, or
  - receipt of S1,F14 that has been improperly formatted or with COMMACK[1] not set to 0.
- A reply timeout period begins after the successful transmission of a complete primary message for which a reply is expected (see SEMI E4 (SECS-I) or SEMI E37 (HSMS) for a protocol-specific definition of reply timeout).

[1] Establish Communications Acknowledge Code, defined in §5.2. See SEMI E5 for further definition of this Data Item.

- A communication link is established following the first successful completion of any one S1,F13/F14 transaction with an acknowledgement of 'accept'. The establishment of this link is logical rather than physical.
- Implementations may have mechanisms which allow outgoing messages to be stored temporarily prior to being sent. The noun queue is used to cover such stored messages. They are queued when placed within the queue and are dequeued by removing them from this storage.
- Send includes 'queue to send' or 'begin the process of attempting to send' a message. It does not imply the successful completion of sending a message.
- The host may attempt to establish communications with equipment at any time due to the initialization of the host or by independent detection of a communications failure by the host. Thus, the host may initiate an S1,F13/F14 transaction at any time.

6.4.2 *CommDelay Timer* — The CommDelay timer represents an internal timer used to measure the interval between attempts to send S1,F13. The length of this interval is equal to the value in the EstablishCommunicationsTimeout. The CommDelay timer is not directly visible to the host.

6.4.2.1 EstablishCommunicationsTimeout is the user-configurable equipment constant that defines the delay, in seconds, between attempts to send S1,F13. This value is used to initialize the CommDelay timer.

6.4.2.2 The CommDelay timer is initialized to begin timing. The CommDelay timer is initialized only when the state WAIT DELAY is entered.

6.4.2.3 The CommDelay timer is expired when it 'times out', and the time remaining in the interval between attempts to send is zero. When the timer expires during the state WAIT DELAY, it triggers a new attempt to send S1,F13 and the transition to the state WAIT CRA.[1]

6.4.3 *Conventions*

- The attempt to send S1,F13 is made only upon transit into the state WAIT CRA. The CommDelay Timer should be set to 'expired' at this time.
- The CommDelay timer is initialized only upon transit into the state WAIT DELAY. A next attempt to send S1,F13 shall occur only upon a transit to the state WAIT CRA.

6.4.4 *Communication States* — There are two major states of SECS communication, DISABLED and ENABLED. The system default state must be user-configurable at the equipment (e.g., via a jumper setting or nonvolatile memory variable).

6.4.4.1 Once system initialization has been achieved, the operator shall be able to change the communication state selection at any time via equipment terminal functions or momentary switch. A two-position type switch must not be used due to possible conflict with the system default.

6.4.4.2 The ENABLED state has two substates, NOT COMMUNICATING and COMMUNICATING, described below. The equipment must inform the operator of the current communication state via continuous display at the equipment, including the NOT COMMUNICATING and COMMUNICATING substates.

6.4.4.3 In the event of a connection transaction failure, a user-configurable equipment constant EstablishCommunicationsTimeout is used to establish the interval between attempts to send an S1,F13 (Establish Communications Request) while in the NOT COMMUNICATING substate.

6.4.4.4 Figure 4 shows the relationship between the superstates and substates of the Communications State Model. A description of the events triggering state transitions and the actions taken is given in Table 1.

[1] CRA is the mnemonic defined for Establish Communications Request Acknowledge (S1,F14).

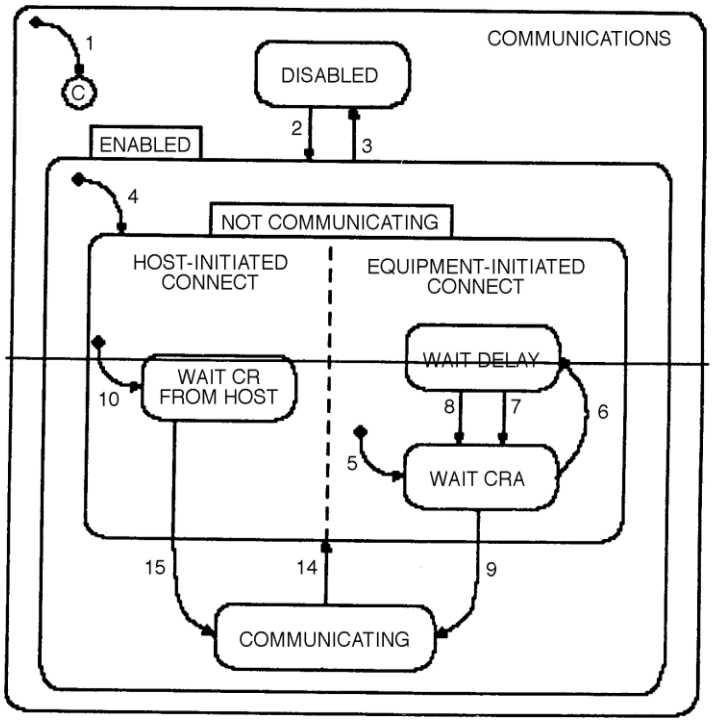



**Figure 5**
**Communications State Diagram**

6.4.4.5 The states of the Communications State Model are defined as follows:

6.4.4.5.1 *DISABLED*

6.4.4.5.1.1 In this state SECS-II communication with a host computer is nonexistent. If the operator switches from ENABLED to DISABLED, all SECS-II communications must cease immediately. Any messages queued to send shall be discarded, and all further action on any open transactions and conversations shall be terminated.[1] Handling of messages currently being transmitted is an issue for lower level message transfer protocols and is not addressed in this Standard.

6.4.4.5.1.2 The DISABLED state is a possible system default.

6.4.4.5.2 *ENABLED*

6.4.4.5.2.1 ENABLED has two substates, COMMUNICATING and NOT COMMUNICATING. Whenever communications are enabled, either during system initialization or through operator selection, the substate of NOT COMMUNICATING is active until communications are formally established. Lower-level protocols (such as SECS-I) are assumed to be functioning normally in that they are capable of supporting the communication of SECS-II syntax.

6.4.4.5.2.2 The ENABLED state is a possible system default.

6.4.4.5.3 *ENABLED/NOT COMMUNICATING*

6.4.4.5.3.1 No messages other than S1,F13, S1,F14, and S9,Fx shall be sent while this substate is active. The equipment shall discard any messages received from the host other than S1,F13 or S1,F14 (Establish Communications Acknowledge). It shall also periodically attempt to establish communication with a host computer by issuing an S1,F13 until communications are successfully established. However, only one equipment-initiated S1,F13 transaction may be open at any time.

[1] Refer to SEMI E5, § 5, for definitions of SECS-II transaction and conversation protocols.

6.4.4.5.3.2 The NOT COMMUNICATING state has two AND substates, HOST-INITIATED CONNECT and EQUIPMENT-INITIATED CONNECT, both of which are active whenever the equipment is NOT COMMUNICATING. These two substates clarify the behavior of the equipment in the event that both the equipment and the host attempt to establish communications during the same period of time.[1]

6.4.4.5.4 *NOT COMMUNICATING/EQUIPMENT-INITIATED CONNECT*

6.4.4.5.4.1 This state has two substates, WAIT CRA and WAIT DELAY. Upon any entry to the NOT COMMUNICATING state, whenever EQUIPMENT-INITIATED CONNECT first becomes active, a transition to WAIT CRA occurs, the CommDelay timer is set to 'expired', and an immediate attempt to send S1,F13 is made.

6.4.4.5.5 *NOT COMMUNICATING/EQUIPMENT-INITIATED CONNECT/WAIT CRA*

6.4.4.5.5.1 An Establish Communications Request has been sent. The equipment waits for the host to acknowledge the request.

6.4.4.5.6 *NOT COMMUNICATING/EQUIPMENT-INITIATED CONNECT/WAIT DELAY*

6.4.4.5.6.1 A connection transaction failure has occurred. The CommDelay timer has been initialized. The equipment waits for the timer to expire.

6.4.4.5.7 *NOT COMMUNICATING/HOST-INITIATED CONNECT*

6.4.4.5.7.1 This state describes the behavior of the equipment in response to a host-initiated S1,F13 while NOT COMMUNICATING is active.

6.4.4.5.8 *NOT COMMUNICATING/HOST-INITIATED CONNECT/WAIT CR FROM HOST*

6.4.4.5.8.1 The equipment waits for an S1,F13 from the host. If an S1,F13 is received, the equipment attempts to send an S1,F14 with COMMACK = 0.

6.4.4.5.9 *ENABLED/COMMUNICATING*

6.4.4.5.9.1 Communications have been established. The equipment may receive any message from the host, including S1,F13. When the equipment is COMMUNICATING, SECS communications with a host computer must be maintained. This state remains active until communications are disabled or a communication failure occurs. If the equipment receives S1,F13 from the host while in the COMMUNICATING substate, it should respond with S1,F14 with COMMACK set to zero. If the equipment receives S1,F14 from a previously sent S1,F13, no action is required.

6.4.4.5.9.2 In the event of communication failure, the equipment shall return to the NOT COMMUNICATING substate and attempt to re-establish communications with the host.

6.4.4.5.9.3 It is possible that the equipment may be waiting for an S1,F14 from the host in EQUIPMENT-INITIATED CONNECT/WAIT CRA at the time an S1,F13 is received from the host in HOST-INITIATED CONNECT/WAIT CR FROM HOST. When this situation occurs, both equipment and host have an open S1,F13/S1,F14 transaction. Since communications are successfully established on the successful completion of any S1,F13/S1,F14 transaction, either of these two transactions may be the first to complete successfully and to cause the transition from NOT COMMUNICATING state to COMMUNICATING. In this event, the other transaction shall remain open regardless of the transition to COMMUNICATING until it is closed in a normal manner.

6.4.4.5.9.4 If the equipment has not yet sent[2] an S1,F14 to a previously received S1,F13 at the time when COMMUNICATING becomes active, the S1,F14 response shall be sent in a normal manner. A failure to send the S1,F14 is then treated as any other communication failure.

6.4.4.5.9.5 If the equipment-initiated S1,F13/S1,F14 is still open when the transition to COMMUNICATING occurs, subsequent failure to receive a reply from the host is considered a communication fault by equipment. An S9,F9 should

[1] Note that in the Harel notation, an exit from any AND substate is an exit from the parent state and thus from all other AND substates of that parent substate.

[2] This includes transmissions that may have started but not yet successfully completed at the time that the transition to COMMUNICATING occurs.

be sent when a transaction timer timeout occurs[1] (see § 5.10 for definitions of communication faults and message faults, as well as detail on Stream 9 Error Messages).

6.4.5 *State Transitions* — Table 1 contains a full description of the state transitions depicted in Figure 4.

6.4.5.1 When the operator switches from the DISABLED state to the ENABLED state, no collection event shall occur, since no messages can be sent until communications have been established. The process of establishing communications serves to notify the host that communications are ENABLED. No other collection events are defined for the Communications State Model.

**Table 1 Communications State Transition Table**

| # | *Current State* | *Trigger* | *New State* | *Action* | *Comment* |
|---|---|---|---|---|---|
| 1 | (Entry to COMMUNICATIONS) | System initialization. | System Default | None. | The system default may be set to DISABLED or ENABLED. |
| 2 | DISABLED | Operator switches from DISABLED to ENABLED. | ENABLED | None. | SECS-II communications are enabled. |
| 3 | ENABLED | Operator switches from ENABLED to DISABLED. | DISABLED | None. | SECS-II communications are prohibited. |
| 4 | (Entry to ENABLED) | Any entry to ENABLED state. | NOT COMMUNICATING | None. | May enter from system initialization to ENABLED or through operator switch to ENABLED. |
| 5 | (Entry to EQUIPMENT-INITIATED CONNECT) | (Any entry to NOT COMMUNICATING) | WAIT CRA | Initialize communications. Set CommDelay timer 'expired'. Send S1,F13. | Begin the attempt to establish communications. |
| 6 | WAIT CRA | Connection transaction failure. | WAIT DELAY | Initialize CommDelay timer. Dequeue all messages queued to send. | If appropriate, dequeued messages shall be placed in spool buffer in the order generated. Wait for timer to expire. |
| 7 | WAIT DELAY | CommDelay timer expired. | WAIT CRA | Send S1,F13 | Wait for S1,F14. May receive S1,F13 from Host. |
| 8 | WAIT DELAY | Received a message other than S1,F13. | WAIT CRA | Discard message. No reply. Set CommDelay timer 'expired'. Send S1,F13. | Indicates opportunity to establish communications. |
| 9 | WAIT CRA | Received expected S1,F14 with COMMACK = 0. | COMMUNICATING | None. | Communications are established. |
| 10 | (Entry to HOST-INITIATED CONNECT) | (Any entry to NOT COMMUNICATING) | WAIT CR FROM HOST | None. | Wait for S1,F13 from Host. |

[1] The existence of a transaction timer is not a requirement in some protocols, such as SMS (SEMI E13).

| # | *Current State* | *Trigger* | *New State* | *Action* | *Comment* |
|---|---|---|---|---|---|
| 14 | COMMUNICATING | Communication failure. (See SEMI E4 or SEMI E37 for a protocol-specific definition of communication failure.) | NOT COMMUNICATING | Dequeue all messages queued to send. | Dequeued messages may be placed in spool buffer as appropriate. |
| 15 | WAIT CR FROM HOST | Received S1,F13. | COMMUNICATING | Send S1,F14 with COMMACK = 0. | Communications are established. |

6.5 *Control State Model* — The CONTROL state model defines the level of cooperation between the host and equipment. It also specifies how the operator may interact at the different levels of host control. While the COMMUNICATIONS state model addresses the ability for the host and equipment to exchange messages, the CONTROL model addresses the equipment's responsibility to act upon messages that it receives.

6.5.1 The CONTROL model provides the host with three basic levels of control. In the highest level (REMOTE), the host may control the equipment to the full extent possible. The middle level (LOCAL) allows the host full access to information, but places some limits on how the host can affect equipment operation. In the lowest level (OFF-LINE), the equipment allows no host control[1] and only very limited information.[2]

6.5.2 The control model and communications model (when implemented) do not interact directly. That is, no action or state of one model directly causes a change in behavior of the other. It is true, however, that when the communication state is NOT COMMUNICATING then most message transaction are not functional. When messages cannot be transmitted, the control capabilities and all other GEM capabilities are affected.

6.5.3 Refer to Figure 5 as the CONTROL substates and state transitions are defined.

6.5.4 *OFF-LINE*

6.5.4.1 When the OFF-LINE state is active, operation of the equipment is performed by the operator at the operator console. While the equipment is OFF-LINE, message transfer is possible. However the use of messaging for any automation purpose is severely restricted. While the OFF-LINE state is active, the equipment will only respond to those messages used for the establishment of communications or a host request to activate the ON-LINE state.

6.5.4.2 While OFF-LINE, the equipment will respond with an Sx,F0 to any primary message from the host other than S1,F13 or S1,F17. It will process and respond to S1,F13 and S1,F17. S1,F17 is used by the host to request the equipment to transition to the ON-LINE state. The equipment will accept this request and send a positive response only when the HOST OFF-LINE state is active (see transition 11 definition below).

6.5.4.3 While the OFF-LINE state is active, the equipment shall attempt to send no primary message other than S1,F13,[3] S9,Fx,[4] and S1,F1 (see ATTEMPT ON-LINE substate). If the equipment receives a reply message from the host other than S1,F14 or S1,F2, this message is discarded.

6.5.4.4 No messages enter the spool when the system is OFF-LINE. Spooling may be active when the Communications State of NOT COMMUNICATING is active. This might occur during OFF-LINE, but since the equipment will not attempt to send messages except as mentioned in the previous paragraph,[5] no messages will enter the spool.

6.5.5 OFF-LINE has three substates: EQUIPMENT OFF-LINE, ATTEMPT ON-LINE, and HOST OFF-LINE.

[1] The host may establish communications. This does not affect equipment operation and for that reason is not termed a control operation.

[2] The host may determine the equipment identification via the S1F13/F14 transaction.

[3] Sending of S1,F13 is based upon the COMMUNICATIONS state model.

[4] S9,Fx messages may be issued only in response to the messages to which the equipment will normally respond while OFF-LINE (i.e., S1,F13 and S1,F17).

[5] The equipment may send S1,F1 or S1,F13, but since Stream 1 messages are not eligible for spooling, they will not enter the spool either.

6.5.6 *OFF-LINE/EQUIPMENT OFF-LINE*

6.5.6.1 While this state is active, the system maintains the OFF-LINE state. It awaits operator instructions to attempt to go ON-LINE.

6.5.7 *OFF-LINE/ATTEMPT ON-LINE*

6.5.7.1 While the ATTEMPT ON-LINE state is active, the equipment has responded to an operator instruction to attempt to go to the ON-LINE state. Upon activating this state, the equipment attempts to send an S1,F1 to the host.

6.5.7.2 Note that when this state is active, the system does not respond to operator actuation of either the ON-LINE or the OFF-LINE switch.

6.5.8 *OFF-LINE/HOST OFF-LINE*

6.5.8.1 While the HOST OFF-LINE state is active, the operator's intent is that the equipment be ON-LINE. However, the host has not agreed. Entry to this state may be due to a failed attempt to go ON-LINE or to the host's request that the equipment go OFF-LINE from ON-LINE (see the transition table for more detail). While this state is active, the equipment shall positively respond to any host's request to go ON-LINE (S1,F17). Such a request shall be denied when the HOST OFF-LINE state is not active.

6.5.9 *ON-LINE*

6.5.9.1 While the ON-LINE state is active, SECS-II messages may be exchanged and acted upon. Capabilities that may be available to the host should be similar to those available from the operator console wherever practical.

6.5.9.2 The use of Sx,F0 messages is not required while the ON-LINE state is active. Their use is discouraged in this case. The only allowed use is to close open transactions in conjunction with message faults.

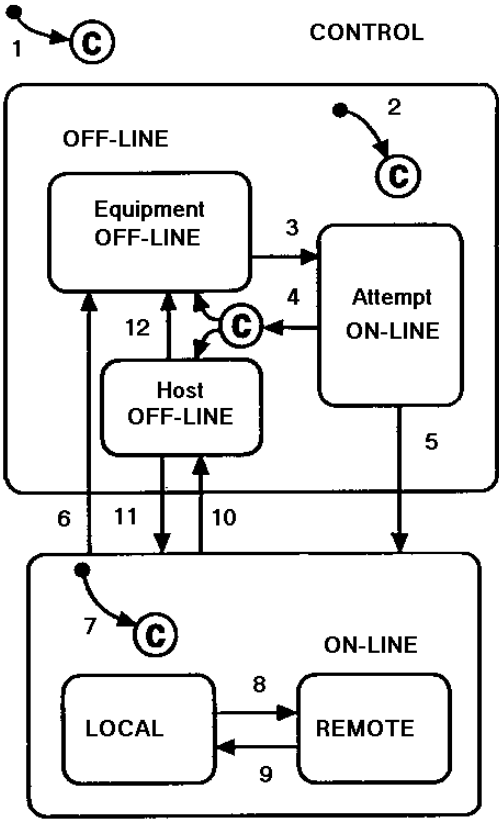



**Figure 6**
**CONTROL State Model**

6.5.10 *ON-LINE/LOCAL*

6.5.10.1 Operation of the equipment is implemented by direct action of an operator. All operation commands shall be available for input at the local operator console of the equipment.

6.5.10.2 The host shall have the following capabilities and restrictions when the LOCAL state is active:

- The host shall be prohibited from the use of remote commands that cause physical movement or which initiate processing. During processing, the host shall be prohibited from the use of any remote command that affects that process.
- During processing, the host shall be prohibited from modifying any equipment constants that affect that process. Other equipment constants shall be changeable during processing. The host shall be able to modify all available equipment constants when no processing is in progress.
- The host shall be capable of initiating the upload and download of recipes to/from the recipe storage area on the equipment. The host shall be capable of selecting recipes for execution so long as this action does not affect any currently executing recipe.
- The host shall be able to configure automatic data reporting capabilities including alarms, event reporting, and trace data reporting. The host shall receive all such reports at the appropriate times.
- The host shall be able to inquire for data from the equipment, including status data, equipment constants, event reports, Process Recipe directories, and alarms.
- The equipment shall be able to perform Terminal Services as defined in GEM.

6.5.10.3 The host shall be allowed any other capabilities that were not specifically restricted in the above items as long as the LOCAL state is active.

NOTE 1: Capabilities mentioned above which are not implemented on a specific equipment may be ignored in this context.

6.5.11 *ON-LINE/REMOTE*

6.5.11.1 For equipment which supports the GEM capability of remote control (see § 5.5), while the REMOTE state is active, the host shall have access, through the communications interface, to the necessary commands to operate the equipment through the full process cycle in an automated manner. The equipment does not restrict any host capabilities when REMOTE is active. The degree of control executed by the host may vary from factory to factory. In some cases, the operator maybe required to interact during remotely controlled processes. This interaction may involve set-up operations, operator assist situations, and others. This state is intended to be flexible enough to accommodate these different situations.

6.5.11.2 To support the different factory automation policies and procedures, it shall be possible to configure the equipment to restrict the operator in specific nonemergency procedures. These restrictions shall be configurable so that the equipment may be set up to allow the operator to perform necessary functions without contention with the host. The categories for configuration shall include (but are not limited to):

- change equipment constants (process-related),
- change equipment constants (non-process-related),
- initiate Process Recipe download,
- select Process Recipe,
- start Process Recipe,
- pause/resume Process Recipe,
- operator assist,
- material movement to/from equipment, and
- equipment-specific commands (on a command-by-command basis if needed).

NOTE 2: Capabilities mentioned above which are not implemented on a specific equipment may be ignored in this context.

LETTER BALLOT

6.5.11.3 No capabilities that are available to the operator when the LOCAL state is active should be unconditionally restricted when the REMOTE state is active. The supplier may provide for configurable restriction of operator capabilities not included in the list above if desired. No configurability is necessary for any operator functions not available to the host.

6.5.11.4 The control functions must be shared to some degree between the host and the local operator. At the very least, the operator must have the capability to change the CONTROL state, actuate an Emergency Stop, and interrupt processing (e.g., STOP, ABORT, or PAUSE). All of these capabilities except Emergency Stop may be access-limited.[1]

6.5.11.5 The host software should be designed to be compatible with the capabilities allotted to the operator.

**Table 2 CONTROL State Transition Table**

| # | *Current State* | *Trigger* | *New State* | *Action* | *Comments* |
|---|---|---|---|---|---|
| 1 | (Undefined) | Entry into CONTROL state (system initialization). | CONTROL (Substate conditional on configuration). | None | Equipment may be configured to default to ON-LINE or OFF-LINE.[#1] |
| 2 | (Undefined) | Entry into OFF-LINE state. | OFF-LINE (Substate conditional on configuration.) | None | Equipment may be configured to default to any substate of OFF-LINE. |
| 3 | EQUIPMENT OFF-LINE | Operator actuates ON-LINE switch. | ATTEMPT ON-LINE | None | Note that an S1,F1 is sent whenever ATTEMPT ON-LINE is activated. |
| 4 | ATTEMPT ON-LINE | S1,F0. | New state conditional on configuration. | None | This may be due to a communication failure,[#2] reply timeout, or receipt of S1,F0. Configuration may be set to EQUIPMENT OFF-LINE or HOST OFF-LINE. |
| 5 | ATTEMPT ON-LINE | Equipment receives expected S1,F2 message from the host. | ON-LINE | None | Host is notified of transition to ON-LINE at transition 7. |
| 6 | ON-LINE | Operator actuates OFF-LINE switch. | EQUIPMENT OFF-LINE | None | 'Equipment OFF-LINE' event occurs.[#3] Event reply will be discarded while OFF-LINE is active. |
| 7 | (Undefined) | Entry to ON-LINE state. | ON-LINE (Substate conditional on REMOTE/LOCAL switch setting.) | None | 'Control State LOCAL' or 'Control State REMOTE' event occurs. Event reported based on actual ON-LINE substate activated. |
| 8 | LOCAL | Operator sets front panel switch to REMOTE. | REMOTE | None | 'Control State REMOTE' event occurs. |
| 9 | REMOTE | Operator sets front panel switch to LOCAL. | LOCAL | None | 'Control State LOCAL' event occurs. |
| 10 | ON-LINE | Equipment accepts 'Set OFF-LINE' message from host (S1,F15). | HOST OFF-LINE | None | 'Equipment OFF-LINE' event occurs. |
| 11 | HOST OFF-LINE | Equipment accepts host request to go ON-LINE (S1,F17). | ON-LINE | None | Host is notified to transition to ON-LINE at transition 7. |
| 12 | HOST OFF-LINE | Operator actuates OFF-LINE switch. | EQUIPMENT OFF-LINE | None | 'Equipment OFF-LINE' event occurs. |

#1 The configuration mentioned for transitions 1 and 2 should be a single setting. This would provide the user with a choice of entering the EQUIPMENT OFF-LINE, ATTEMPT ON-LINE, HOST OFF-LINE, or ON-LINE states.

#2 Communication failures are protocol specific. Refer to the appropriate protocol Standard (e.g., SEMI E4 or SEMI E37) for a protocol-specific definition of communication failure.

[1] Definition of the method of limiting operator access (password, key, etc.) to a capability is not within the scope of this Document.

#3 Any host initiated transaction open at the equipment must be completed. This may happen either by sending the appropriate reply to the host prior to sending the event message or by sending an Sx,F0 message following the event message (i.e., after the transaction).

6.6 *Equipment Processing States* — The behavior of the equipment in the performance of its intended function must be documented. This processing state model is highly dependent on the equipment process, technology, and style. However, there are expected to be common aspects to these models.

6.6.1 The Processing State Diagram, Figure 6, is provided as an example of an implementation model. This model demonstrates the expected nature of the processing state model documentation. There is no requirement that these specific states be implemented.

6.6.2 The equipment must generate collection events for each processing state transition, as well as provide status variables (ProcessState, PreviousProcessState) whose values are the current processing state and the previous processing state.

6.6.3 In referring to the Processing State Diagram, note that the initialization state INIT is not an actual processing state. It is shown here simply to indicate that the IDLE processing state is entered upon completion of equipment system initialization. On the following pages detailed descriptions are provided for the equipment processing states and state transitions (numbered) as shown in the diagram.

6.6.4 *Description of Equipment Processing States*

6.6.4.1 *IDLE*

6.6.4.1.1 In this state the equipment is awaiting instructions.

6.6.4.2 *PROCESSING ACTIVE*

6.6.4.2.1 This state is the parent of all substates where the context of Process Recipe execution exists.

6.6.4.3 *PROCESS*

6.6.4.3.1 This state is the parent of those substates that refer to the active preparation and execution of a process program.

6.6.4.4 *SETUP*

6.6.4.4.1 In this state all external conditions necessary for process execution are satisfied, such as ensuring material is present at the equipment, input/output ports are in the proper state, parameters such as temperature and pressure values are within limits, etc. If all setup operations are already complete, then this becomes a fall through state and a transition to the next state takes place.

6.6.4.5 *READY*

6.6.4.5.1 In this state the equipment is ready for process execution and is awaiting a START command from the operator or the host.

6.6.4.6 *EXECUTING*

6.6.4.6.1 Executing is the state in which the equipment is executing a Process Recipe automatically and can continue to do so without external intervention.

6.6.4.7 *PAUSE*

6.6.4.7.1 In this state processing is suspended and the equipment is awaiting a command.

6.6.4.7.2 Each state transition is defined in the following table. Note that all transitions in this table should be considered collection events.

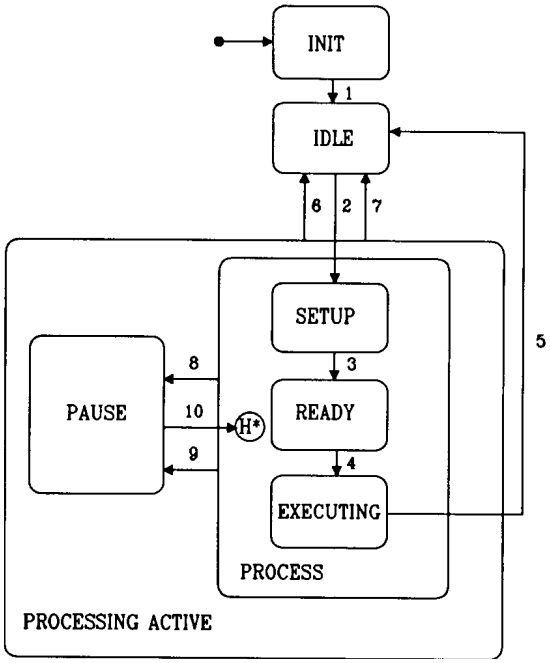



**Figure 7**
**Processing State Diagram**

**Table 3 Processing State Transition Table**

| # | *Current State* | *Trigger* | *New State* | *Action* | *Comments* |
|---|---|---|---|---|---|
| 1 | INIT | Equipment initialization complete. | IDLE | None. | None. |
| 2 | IDLE | Commit has been made to set up. | SETUP | None. | None. |
| 3 | SETUP | All setup activity has completed and the equipment is ready to receive a START command. | READY | This activity is equipment-specific. | None. |
| 4 | READY | Equipment has received a START command from the host or operator console. | EXECUTING | This activity is equipment-specific. | None. |
| 5 | EXECUTING | The processing task has been completed. | IDLE | None. | None. |
| 6 | PROCESSING ACTIVE | Equipment has received a STOP command from host or operator console. | IDLE | None. | None. |
| 7 | PROCESSING ACTIVE | Equipment has received an ABORT command from host or operator console. | IDLE | This activity is equipment-specific. | None. |
| 8 | PROCESS | The equipment decides to PAUSE due to a condition such as alarm. | PAUSE | This activity is equipment-specific. | For this type of problem, an operator assist is usually required. |
| 9 | PROCESS | Equipment has received a PAUSE command from host or operator console. | PAUSE | This activity is equipment-specific. | None. |
| 10 | PAUSE | Equipment has received a RESUME command from host or operator console. | Previous PROCESS substate | This activity is equipment-specific. | None. |

LETTER BALLOT

## 7 Equipment Capabilities and Scenarios

7.1 This section describes the details of the capabilities required by GEM and provides scenarios for their use. Capabilities are operations performed by semiconductor manufacturing equipment. These operations are initiated through the communications interface using SECS-II messages. A scenario is a group of SECS-II messages arranged in a sequence to perform a capability. Other information may be included with the scenario for clarity. For each capability, the reader is provided with a statement of purpose, pertinent definitions, a detailed description, requirements, and scenarios.

7.1.1 The following capabilities are discussed:

- Establish Communications
- Event Notification
- Dynamic Event Report Configuration
- Variable Data Collection
- Trace Data Collection
- Limits Monitoring
- Status Data Collection
- On-line Identification
- Alarm Management
- Remote Control
- Equipment Constants
- Process Recipe Management
- Material Movement
- Equipment Terminal Services
- Error Messages
- Clock
- Spooling
- Control

7.2 *Establish Communications* — The Establish Communications capability provides a means of formally establishing communications following system initialization or any loss of communications between communicating partners, and thus of notifying the communication partner that a period of noncommunication has occurred.

7.2.1 *Purpose* — Communications between host and equipment are formally established through use of the Establish Communications Request/Establish Communications Acknowledge transaction.

7.2.1.1 The use of S1,F1/F2 for this purpose is ambiguous since the transaction can be used for other purposes and may occur at any time.

7.2.1.2 The S1,F13/F14 transaction, used in conjunction with the Communications State Model, provides a means for equipment to notify the host, or the host to notify the equipment, that there has been a period of inability to communicate. The successful completion of this transaction also signals a possible need for synchronization activities between host and equipment.

7.2.2 *Definitions*

7.2.2.1 *COMMACK*— Acknowledge code returned in the Establish Communications Acknowledge message. See SEMI E5 for a full definition of this data item.

LETTER BALLOT

7.2.2.2 *EstablishCommunicationsTimeout* — An equipment constant used to initialize the interval between attempts to resend an Establish Communications Request. This value specifies the number of seconds for the interval. See SEMI E5 for a full definition of this variable data item.

7.2.3 *Description* — There are potential problems when one side of the communications link fails and the other side does not detect it. From the point of view of the host, a loss of communications has many possible causes. In some cases, host-controlled settings on the equipment may need to be reset. In other cases, the equipment may have continued an automatic processing sequence during the period of no communication and may have changed states. The definition of a formal protocol for establishing communications alerts the host to the need to synchronize itself with the equipment's current status.

7.2.3.1 Equipment shall consider communications as formally established whenever either of the following conditions have been satisfied:[1]

- Communications Request has been sent to the host and an Establish Communications Acknowledge has been received within the transaction timeout period and with an acknowledge code of Accept, or
- Communications Request has been received from the host, and an Establish Communications Acknowledge response has been successfully sent with an acknowledge code of Accept.

7.2.3.2 When the equipment sends an Establish Communications Request to the host, this notifies the host of the possible need to synchronize itself with the equipment.

7.2.3.3 When the equipment is attempting to establish communications, an Establish Communications Request shall be sent periodically until communications have been formally established as described above. The interval between attempts must be user-configurable and begins as soon as a connection transaction failure is detected (see § 4.4).

7.2.3.4 Attempting to establish communications is not a low-level connectivity issue, but rather a logical application issue used by either party to notify its partner that the host may need to perform synchronization activities with the equipment.

7.2.4 *Requirements*

- Equipment must support the Communication State Model (see § 4.4).
- Equipment must provide the EstablishCommunicationsTimeout equipment constant described above.

7.2.5 *Scenarios*

7.2.5.1 *Host Attempts to Establish Communications*

| COMMENT | HOST | EQUIPMENT | COMMENT |
|---|---|---|---|
| | | | Communications state is Enabled (any substate) |
| Establish Communications Request | S1,F13--> | | |
| | | <--S1,F14 | Reply COMMACK = Accept and Communications state = COMMUNICATING |

[1] Satisfaction of either of these conditions will result in a transition to the COMMUNICATING substrate. See § 4.4 for further detail.

LETTER BALLOT

7.2.5.2 *Equipment Attempts to Establish Communications and Host Acknowledges*

| COMMENT | HOST | EQUIPMENT | COMMENT |
|---|---|---|---|
| | | Communications State = NOT COMMUNICATING | |
| | | | [LOOP] |
| | | | [LOOP]--SEND |
| | | <--S1,F13 | Establish Communications Request |
| Establish Communications Acknowledge | S1,F14--> | | |
| | | | [IF] S1,F14 rcvd w/o timeouts |
| | | | [THEN] exit_loop--SEND |
| | | | [ELSE] Delay for interval in Establish Commuications-Timeout |
| | | | [ENDIF] |
| | | | [END_LOOP]--SEND |
| | | | [IF] COMMACK = Accept |
| | | | [THEN] Communications state= Communicating exit_loop-- |
| | | | [ELSE] Reset timer for delay, and delay for interval specified in EstablishCommunications—Timeout |
| | | | [ENDIF] |
| | | | [END_LOOP] |

7.2.5.3 *Simultaneous Attempts to Establish Communications* — For equipment that supports interleaving, it is possible that either the host or equipment could send an Establish Communications Request before receiving the request from its partner. As communications are established by the successful acceptance of any one Establish Communications Request, it is immaterial who sends the request first. The roles of host and equipment may be reversed.

7.2.5.3.1 Equipment Receives S1,F14 From Host Before Sending S1,F14:

| COMMENT | HOST | EQUIPMENT | COMMENT |
|---|---|---|---|
| | | Communications State = NOT COMMUNICATING | |
| | | <--S1,F13 | Establish Communications Request |
| Establish Communications Request | S1,F13--> | | |
| Reply COMMACK = Accept | S1,F14--> | | S1,F14 received from Host and Communications established[1] and Communications state = COMMUNICATING |
| | | <--S1,F14 | Reply COMMACK = Accept[2] |

[1] Communications are established at the successful completion of the S1,F13/14 transaction where COMMACK is set to zero.

[2] Communications are established on the successful transmission of S1,F14, even if there is an open S1,F13.

LETTER BALLOT

7.2.5.3.2 Equipment Sends S1,F14 To Host Before Receiving S1,F14:

| COMMENT | HOST | EQUIPMENT | COMMENT |
|---|---|---|---|
| | | Communications State = NOT COMMUNICATING | |
| | | <--S1,F13 | Establish Communications Request |
| Establish Communications Request | S1,F13--> | | |
| | | <--S1,F14 | Reply COMMACK = Accept[15] Communications established[15] and Communications state = COMMUNICATING |
| Reply COMMACK = Accept | S1,F14--> | | S1,F14 received from Host |

7.3 *Data Collection* — Data collection allows the host to monitor equipment activity via event reporting, trace data reporting, limits monitoring, and query of selected status or other variable data.

7.3.1 *Event Data Collection* — Event data collection provides a dynamic and flexible method for the user to tailor the equipment to meet individual needs with respect to data representation and presentation to the host. The event-based approach to data collection provides automatic notification to the host of equipment activities and is useful in monitoring the equipment and in maintaining synchronization with the equipment.

7.3.1.1 Event data collection may be broken into two logical parts: host notification when an event occurs, and dynamic configuration of the data attached to the event notification.

7.3.1.2 *Event Notification* — This section describes the method of notifying the host when equipment collection events occur.

7.3.1.2.1 *Purpose* — This capability provides data to the host at specified points in equipment operation. This asynchronous reporting eliminates the need for the host to poll the equipment for this information. Events on the equipment may trigger activity on the part of the host. Also, knowledge of the occurrence of events related to the equipment state models allows the host to track the equipment state. An equipment's behavior is related to its current state. Thus, this capability helps the host understand how an equipment will behave and how it will react to host behavior.

7.3.1.2.2 *Definitions*

7.3.1.2.2.1 *Collection Event* — An event (or grouping of related events) on the equipment that is considered significant to the host.

7.3.1.2.2.2 *Collection Event ID (CEID)* — A unique identifier of a collection event. See SEMI E5 for a full definition of this data item.

7.3.1.2.2.3 *Event* — A detectable occurrence significant to the equipment.

7.3.1.2.2.4 *Report* — A set of variables predefined by the equipment or defined by the host via S2,F33/F34.

7.3.1.2.3 *Detailed Description* — The equipment supplier must provide a set of predefined collection events. Specific collection events are required by individual capabilities and state models. Examples of collection events include:

- The completion of each action initiated by a host requested command,
- Selected processing and material handling activities,
- Operator action detected by the equipment,
- A state transition,
- The setting or clearing of an alarm condition on the equipment, and
- Exception conditions not considered alarms.

LETTER BALLOT

7.3.1.2.3.1 See § 7 for a list of required collection events.

7.3.1.2.3.2 The reporting of a collection event may be disabled per event by the user to eliminate unwanted messages. An event report message shall be sent to the host upon the occurrence of a particular collection event if the collection event (CEID) has been enabled. Attached to each event message is one or more event reports which contain variable data. § 5.3.1.3 describes the capability which allows for the dynamic customization of event reports. The values of any data contained in an event report message must be current upon the occurrence of the event. This implies that event reports be built at the time of the event occurrence.

7.3.1.2.3.3 The equipment shall also provide the S6,F15/F16 transaction to allow the host to request the data from a specific event report.

7.3.1.2.4 Requirements

1. The equipment supplier shall provide documentation of all collection events defined on the equipment and the conditions for each event to occur.
2. The equipment supplier shall provide unique CEIDs for each of the various collection events that are available for reporting.
3. The equipment supplier shall provide a method for enabling and disabling the reporting of each event. This method shall either be available via the host interface (see § 5.3.1.3) or the equipment's operator console.
4. For each event, the equipment supplier shall provide either:
   - a default set of report(s) linked to the event which contain data pertinent to that event, or
   - the ability for the user to configure the data linked to that event via the equipment's operator console or host interface (see § 5.3.1.3).

7.3.1.2.5 *Scenarios*

7.3.1.2.5.1 Collection Event Occurs on the Equipment:

```
COMMENT                              HOST    EQUIPMENT    COMMENT

                                                          [IF] Event Report is Multi-block
                                             <--S6,F5     [THEN] send Multi-block inquire
Multi-block grant                    S6,F6-->
                                                          [ENDIF]
                                             <--S6,F11    Equipment sends Event Report
Host acknowledges Event Report       S6,F12-->
```

7.3.1.2.5.2 Host Requests Event Report:

```
COMMENT                              HOST    EQUIPMENT    COMMENTS

Host requests an event report        S6,F15-->
                                             <--S6,F16    Equipment sends event report.
```

7.3.1.3 *Dynamic Event Report Configuration* — This section describes a capability which allows the host to dynamically modify the equipment event reporting setup.

7.3.1.3.1 *Purpose* — This capability is defined to provide the data reporting flexibility required in some manufacturing environments. It allows the host to increase or decrease the data flow according to need. For example, if the performance of an equipment degrades, the data flow from that equipment may be increased to help diagnose the problem.

7.3.1.3.2 *Definitions*

7.3.1.3.2.1 *EventsEnabled* — A variable data item that consists of a list of currently enabled collection events (CEIDs). See SEMI E5 for a full definition of this variable data item.

7.3.1.3.2.2 *Report ID (RPTID)* — A unique identifier of a specific report. See SEMI E5 for a full definition of this data item.

7.3.1.3.2.3 *Variable Data (V)* — A data item containing status (SV), data (DVVAL), or constant (ECV) values. See SEMI E5 for a full definition of this data item.

7.3.1.3.2.4 *Variable Data ID (VID)* — A unique identifier of a variable data item. The set of VID's is the union of all SVID's, ECID's, and ID's for DVVAL's (DVNAME's). See SEMI E5 for a full definition of this data item.

7.3.1.3.3 *Detailed Description* — The equipment shall support the following event report configuration functionality through the SECS-II interface:

- Host definition/deletion of custom reports,
- Host linking/unlinking of defined reports to specified collection events, and
- Host enabling/disabling the reporting of specified collection events.

NOTE 3: The equipment may also supply alternative means for defining reports and linking reports to events (e.g., via the operator console). Implementation of alternate means is not required.

7.3.1.3.3.1 The equipment can be instructed by the host to enable or disable reporting of collection events on an individual or collective basis. A status value (SV) shall be available that consists of a list of enabled collection events (see § 6.4, Variable Item List, EventsEnabled variable).

7.3.1.3.3.2 Reports may be attached to an event report message (S6,F11). These reports are specifically linked to the desired event and typically contain variable data relating to that event. The reports may be provided by the equipment supplier or created by the user. The user must be able to create reports and link them to events via the SECS-II interface.

7.3.1.3.3.3 The data reported in the event report messages may consist of Status Values (SV's), Equipment Constant Values (ECV's), or Data Values (DVVAL's). Note that data values shall be valid and current on certain events and certain states and might not be current at other times. The implementor shall document when a data value will be current and available for reporting.

7.3.1.3.4 *Requirements*

- The equipment manufacturer must provide documentation of all variable data available from the equipment. This is to include variable name, variable type or class (SV, ECV, DVVAL), units, format codes, possible range of values, and a description of the meaning and use of this variable.
- The equipment manufacturer must provide unique VIDs for the various variable data (V) available for data collection in the equipment. For example, this means that no SV shall have a VID which is the same as the VID of any ECV or DVVAL.
- All variable data must be available for report definition and event data collection. See § 6.4, Variable Item List, for a list of required variable data.
- All report definitions, report-to-event links, and enable/disable status of event reports must be retained in nonvolatile storage.

LETTER BALLOT

7.3.1.3.5 *Scenario*

7.3.1.3.5.1 Collection Event Reporting Set-up:

| COMMENT | HOST | EQUIPMENT | COMMENT |
|---|---|---|---|
| [IF] Define Report is Multi-block | | | |
| [THEN] send Multi-block inquire | S2,F39--> | | |
| | | <--S2,F40 | Multi-block grant. |
| [ENDIF] | | | |
| Send report definitions | S2,F33--> | | DATAIDs, RPTIDs, and VIDs received. |
| | | <--S2,F34 | DRACK[1] = 0 the reports are OK |
| [IF] Link Event/Report is Multi-block | | | |
| [THEN] send Multi-block inquire | S2,F39--> | | |
| | | <--S2,F40 | Multi-block grant. |
| [ENDIF] | | | |
| Link reports to events | S2,F35--> | | CEIDs and the corresponding RPTIDs are received. |
| | | <--S2,F36 | LRACK = 0 the event linkages are acceptable. |
| Enable specific collection events | S2,F37--> | | Enable/ disable codes (CEEDs) and the respective event reporting CEIDs received. |
| | | <--S2,F38 | ERACK = 0 OK, will generate the specified reports when the appropriate collection events happen |

7.3.1.4 *Data Variable and Collection Event Namelist Request* — This section describes a method for the host to query valid event data collection configuration from the equipment.

7.3.1.4.1 *Purpose* — Data variable and collection event namelist requests allow the host to query the equipment for existing data variables and collection events as well as their dependencies in order to create valid event reports.

7.3.1.4.2 *Detailed Description* — The equipment shall support the following data variable and event namelist requests through the SECS-II interface:

- Data Variable Namelist Request (DVNR)
- Collection Event Namelist Request (CENR)
  - The CENR includes reporting of the dependencies between data variables and events.

7.3.1.4.3 *Requirements*

5. The equipment shall support the message S1,F21 Data Variable Namelist Request (DVNR) to enable the host to query data variables from the equipment.
6. The equipment shall support the message S1,F23 Collection Event Namelist Request (CENR) to enable the host to query collection events including their associated data variables from the equipment.

[1] Define Report Acknowledge Code, see SEMI E5 for full definition of this Data Item.

LETTER BALLOT

7.3.1.4.4 *Scenarios*

7.3.1.4.4.1 Host initiates: Data Variable Namelist Request (DVNR):

| COMMENTS | HOST | EQUIPMENT | COMMENTS |
|---|---|---|---|
| Host initiates DVNR | S1,F21--> | | |
| | | <--S1,F22 | Data Variable Namelist (DVN) |

7.3.1.4.4.2 Host initiates: Collection Event Namelist Request (CENR):

| COMMENTS | HOST | EQUIPMENT | COMMENTS |
|---|---|---|---|
| Host initiates DVNR | S1,F23--> | | |
| | | <--S1,F24 | Collection Event Namelist (CEN) |

7.3.2 *Variable Data Collection*

7.3.2.1 *Purpose* — This capability allows the host to query for the equipment data variables and is useful during initialization and synchronization.

7.3.2.2 *Definitions*

7.3.2.2.1 *Report ID (RPTID)* — A unique identifier of a specific report. See SEMI E5 for a full definition of this data item.

7.3.2.2.2 *Variable Data (V)* — A variable data item containing status, discrete, or constant data. See SEMI E5 for a full definition of this data item.

7.3.2.3 *Detailed Description* — The host may request a report containing data variables from the equipment by specifying the RPTID. It is assumed that the report has been previously defined (e.g., using the Define Report S2,F33 transaction [see § 5.3.1]). The values of any status variables (SV's) and equipment constants (ECV's) contained within the report must be current. Discrete data values (DVVAL's) are only guaranteed to be valid upon the occurrence of a specific collection event. If DVVAL cannot be specified in equipment due to some restrictions depend on hardware and/or software conditions, the zero length item is reported.

7.3.2.4 *Requirements*

- Variable data items (V's) and associated units of measure must be provided by the equipment manufacturer.

7.3.2.5 *Scenario*

7.3.2.5.1.1 Host Requests Report:

| COMMENT | HOST | EQUIPMENT | COMMENT |
|---|---|---|---|
| Host requests data variables contained in report RPTID | S6,F19--> | | |
| | | <--S6,F20 | Equipment responds with a list of variable data for the given RPTID. |

7.3.3 *Trace Data Collection*

7.3.3.1 *Purpose* — Trace data collection provides a method of sampling data on a periodic basis. The time-based approach to data collection is useful in tracking trends or repeated applications within a time window, or monitoring of continuous data.

7.3.3.2 *Definitions*

7.3.3.2.1 *Data Sample Period (DSPER)* — The time delay between samples. See SEMI E5 for a full definition of this data item.

LETTER BALLOT

7.3.3.2.2 *Reporting Group Size (REPGSZ)* — The number of samples included per trace report transmitted to the host. See SEMI E5 for a full definition of this data item.

7.3.3.2.3 *Status Variable (SV)* — Status data item (included in trace report). See SEMI E5 for a full definition of this data item.

7.3.3.2.4 *Status Variable ID (SVID)* — A unique identifier of a status variable. See SEMI E5 for a full definition of this data item.

7.3.3.2.5 *Total Samples* (TOTSMP) — Number of samples to be taken during a complete trace period. See SEMI E5 for a full definition of this data item.

7.3.3.2.6 *Trace Request ID (TRID)* — An identifier associated with a trace request definition. See SEMI E5 for a full definition of this data item.

7.3.3.3 *Detailed Description* — The equipment shall establish a trace report as instructed by the host (S2,F23). For a trace report (S6,F1), the host shall designate a name for the trace report (TRID), a time interval for data sampling (DSPER), the total number of samples to be taken (TOTSMP), the number of samples per trace report (REPGSZ), and a listing of which data will be sent with the report (SVID's). The number of trace reports sent to the host is determined by total samples divided by reporting group size (TOTSMP/REPGSZ).

7.3.3.3.1 The equipment shall sample the specified data (SV's) at the interval designated by the host (DSPER) and shall send a predefined trace report to the host for the specified reporting group size (REPGSZ). The trace report definition shall be automatically deleted from the equipment after the last trace report has been sent.

7.3.3.3.2 The host may modify or re-initiate a trace function currently in progress by specifying the same TRID in a trace request definition, at which point the old trace shall be terminated and the new trace shall be initiated, or the host can instruct the equipment to terminate a trace report prior to its completion by specifying TOTSMP = 0 for that TRID, at which point the trace report definition shall be deleted.

7.3.3.3.3 A detailed example is included as Related Information § R1-4 ~~Application Note § A.4~~.

7.3.3.4 *Requirements*

- The equipment must have a local mechanism (e.g., internal clock) for triggering the periodic sampling and transmission of trace reports to the host.
- A minimum of four concurrent traces shall be supported by the equipment. The same SVID may be collected in multiple traces simultaneously.
- All SVID's available at the equipment shall be supported for trace data collection. The exception to this is any SV that will not fit into a single block.

NOTE 4: SEMI E5 provides for SV's to be of a list format. Since this may in practice be a variable list, there is a potential problem with such an SV supported by the Trace Data Collection capability. This is a problem with the SEMI E5 Standard. Care should be exercised in the use of SV's using the list format.

LETTER BALLOT

7.3.3.5 *Scenario*

7.3.3.5.1 Host Initiates Trace Report:

| COMMENT | HOST | EQUIPMENT | COMMENT |
|---|---|---|---|
| Trace Data initialization requested | S2,F23--> | | |
| | | <--S2,F24 | Acknowledge, trace initiated |
| | | | [DO] TOTSMP REPGSZ times |
| | | | [DO] REPGSZ many times: collect $SVID_1$,...$SVID_n$ data, delay time by DSPER. |
| | | | [END_DO] |
| | | <--S6,F1 | Send $SV_1$,...$SV_n$ |
| Acknowledge receipt | S6,F2--> | | |
| | | | [END_DO] |
| Optional: | | | |
| Request trace termination prior to completion (TOTSMP = 0) | S2,F23--> | | |
| | | <--S2,F24 | Acknowledge premature termination |

7.3.4 *Limits Monitoring* — This capability relates to the monitoring of selected equipment variables and has three primary aspects:

- Defines a standard set of monitoring zones and limits.
- Provides for reporting to the host when selected equipment variables transition between monitoring zones.
- Empowers the host to modify the values of the variable limit attributes for these same selected equipment variables.

7.3.4.1 *Purpose* — The limits monitoring capability provides the host a means of monitoring equipment conditions by a flexible, efficient, and asynchronous method which is consistent across equipment. It eliminates the need for constant polling of equipment by the host for current status values. Further, this capability allows the host to implement changes in the monitoring range as needed. This capability has application to both production operation and diagnostic/testing scenarios, and it also has applicability to statistical process control.

7.3.4.2 *Definitions*

7.3.4.2.1 *LimitVariable* — DVVAL containing the VID of a specific equipment variable for which a zone transition collection event has been generated.

7.3.4.2.2 *EventLimit* — DVVAL containing the LIMITID of the limit crossed by LimitVariable.

7.3.4.2.3 *TransitionType* — DVVAL which defines the direction of the zone transition which has occurred: 0 = transition from lower to upper zone, 1 = transition from upper to lower zone.

7.3.4.2.4 *Limit* — Used in this section to represent the set of variable limit attributes that completely describe a variable monitoring 'barrier'. The attributes include VID, Units, UPPERDB, LOWERDB, LIMITMAX, and LIMITMIN. In some contexts it may be interpreted more narrowly as the combination of UPPERDB and LOWERDB.

7.3.4.2.5 *$LIMITID_n$* — Refers to the identifier of a specific limit (as defined by UPPERDB and LOWERDB) among the set of limits for a monitored equipment variable. LIMITIDs are consecutively numbered, beginning at one through the number of limits possible (seven minimum).

7.3.4.2.6 *Monitoring Zone* — A subset of the possible range of values for a variable of interest to the host. A single limit divides the range into two zones. Multiple limits may be combined to divide the range even further.

7.3.4.2.7 *Zone Transition* — The movement of a variable value from one monitoring zone to another. This transition is a collection event and has a corresponding CEID.

LETTER BALLOT

7.3.4.2.8 *Deadband* — An overlap of two zones implemented to prevent constant zone transitions by a variable sitting on or near a limit (i.e., 'chattering').

7.3.4.2.9 *UPPERDB* — A variable limit attribute that defines the upper boundary of the deadband of a limit.[1] The value applies to a single limit (LIMITID) for a specified VID. Thus, UPPERDB and LOWERDB as a pair define a limit.

7.3.4.2.10 *LOWERDB* — A variable limit attribute that defines the lower boundary of the deadband of a limit.[18] The value applies to a single limit (LIMITID) for a specified VID. Thus, UPPERDB and LOWERDB as a pair define a limit.

7.3.4.2.11 *UPPER ZONE* — The range of values lying above a limit.

7.3.4.2.12 *LOWERZONE* — The range of values lying below a limit.

7.3.4.2.13 *LIMITMAX* — The maximum value for any limits of a specific equipment variable. This value is set by the equipment manufacturer and typically coincides with the maximum value allowed for the monitored variable.

7.3.4.2.14 *LIMITMIN* — The minimum value for any limits of a specific equipment variable.[19] This value is set by the equipment manufacturer and typically coincides with the minimum value allowed for the monitored variable.

7.3.4.2.15 *Undefined* — When used in reference to variable limits, it indicates that monitoring/reporting of zone transitions involving that particular limit are disabled.

7.3.4.3 *Description* — The limits monitoring capability provides the host with a minimum of seven configurable limits or barriers that may be applied to selected equipment status variables (SV's) of the types floating point, integer, and boolean. When one of these barriers is crossed, a collection event is generated to alert the host to a change in monitoring zone or state of the monitored variable. These seven limits may be combined in a variety of ways to match the needs of the host system.[2] An illustration of a combination of five of the limits to provide one type of variable monitoring is shown in Figure 7.[3] This section describes the key aspects of limits monitoring. Detailed implementation examples of limits monitoring are provided as Related Information § R1-7 ~~Application Note § A.7~~.

NOTE 5: While the SEMI E5 Standard allows SV's to be lists, such variable lists are not allowed under this capability.

7.3.4.3.1 *Monitoring Limit Characteristics* — A limit is defined by a set of attributes that include the variable (VID) to which the limit corresponds, the units of that variable, the maximum and minimum possible values of the limit (LIMITMAX and LIMITMIN) and the specific borders of the limit (UPPERDB and LOWERDB). See Figure 8. There is a limitation to the values of UPPERDB and LOWERDB which may be stated as:

- LIMITMAX≥UPPERDB≥LOWERDB≥LIMITMIN

7.3.4.3.1.1 A limit divides the possible range of variable values into two parts, the upper zone and the lower zone. At any time, the monitored variable is considered to be in one and only one of these zones. However, as Figure 8 shows, these two zones have an area of overlap. This is called the deadband.

[1] The format and units must be the same as the format of the variable being monitored.

[2] Note that while at least seven limits per variable are available from the equipment, the host need not use all seven.

[3] This illustration shows the reading which might be available to the equipment, not the limit excursions reported to the host. Reporting is covered later in the section.

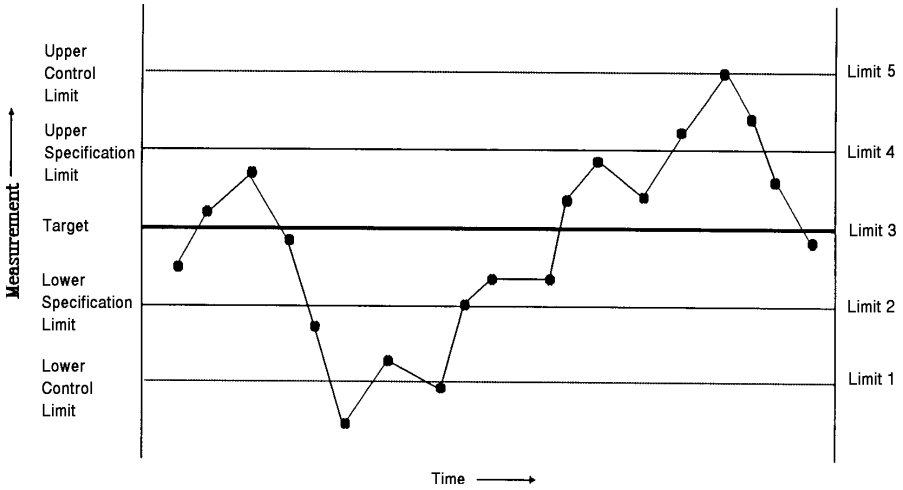



**Figure 8**
**Limit Combination Illustration: Control Application**

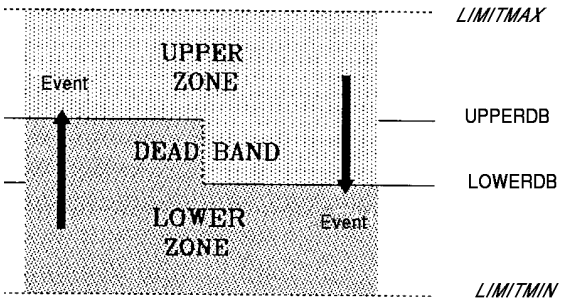



**Figure 9**
**Elements of One Limit**

7.3.4.3.1.2 The deadband is a key concept of limits monitoring, especially for floating point variables. Its purpose is to prevent a phenomenon known as chattering—the repeated changing of zones due to small, rapid fluctuations in variable value while near the zone boundary. In practice, the value of a variable must reach the opposite boundary of the deadband before a zone transition can occur. Thus, if a variable's value reaches the UPPERDB and transitions into the upper zone, it will not return to the lower zone until it falls back to the LOWERDB. The difference between UPPERDB and LOWERDB should always be greater than the typical amplitude of those fluctuations deemed to be insignificant. In some cases, the width of the deadband may set to zero (i.e., UPPERDB = LOWERDB). At first glance, this would seem to make indeterminate the current zone when an integer value sits on the limit. This is not the case, however, when movement of the value is considered. To illustrate, an example is given, assuming that UPPERDB = LOWERDB = 100. The list shows consecutive readings of the variable and the resultant zone:

- 99 Lower Zone (Initial Reading)
- 101 Upper Zone (Zone Transition)
- 100 Lower Zone (Zone Transition)
- 100 Lower Zone
- 99 Lower Zone
- 100 Upper Zone (Zone Transition)

7.3.4.3.1.3 Transition from one zone into another generates a collection event, as might be reported via S6,F11. The host has the option of receiving notification by enabling event reporting for the event. For each variable that has monitoring capability, one CEID is reserved to indicate zone transitions for that variable. To aid in the determination of the nature of a transition event, three DVVAL's have been defined:

- *LimitVariable* — The VID of the monitored variable to which the collection event refers.
- *EventLimit* — Contains the LIMITID of the limit reached or crossed by LimitVariable.
- *TransitionType* — Defines the direction of the zone transition which has occurred: 0 = transition from lower to upper zone, 1 = transition from upper to lower zone.

7.3.4.3.1.4 Sampling frequency is an important element of limits monitoring and should be considered during equipment specification. If changes in variable value are relatively fast compared to sampling frequency, it is possible for some zone transitions to be missed or for multiple zone transitions to occur between readings. Since it is possible for zone transitions to occur 'simultaneously' or for limits to be identically defined, the DVVAL EventLimit has been defined to allow for a list of multiple zone transitions of a variable to be reported with a single collection event.

7.3.4.3.1.5 It also should be emphasized that a single CEID is used to report transitions in both directions across a limit. Thus, reporting for one direction but not the other cannot be configured.

7.3.4.3.1.6 The functionality of each limit for each variable can be described with the state model shown in Figure 9. Below, the three states are described more fully, followed by a table defining the transitions.

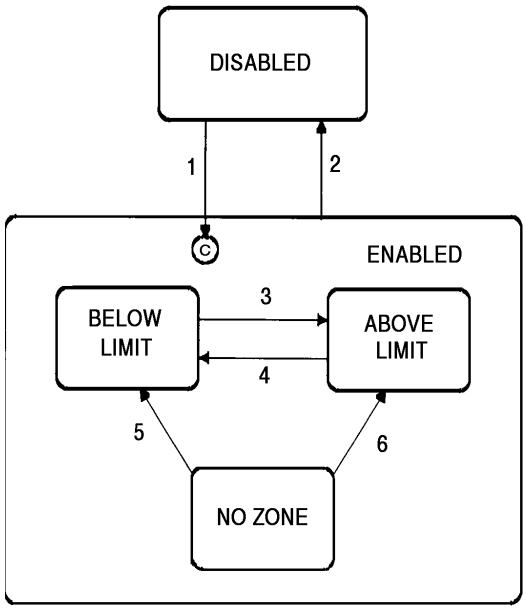



**Figure 10**
**Limit State Model**

7.3.4.3.1.7 *ABOVE LIMIT*

7.3.4.3.1.7.1 A variable is considered to be above a limit when its value increases to equal or exceed the upper boundary of the deadband, UPPERDB. The significance attached to this state is a function of the host's usage.

7.3.4.3.1.8 *BELOW LIMIT*

7.3.4.3.1.8.1 A variable is considered to be below a limit when its value decreases to equal or fall below the lower boundary of the deadband, LOWERDB. The significance attached to this state is a function of the host's usage.

7.3.4.3.1.9 *NO ZONE*

7.3.4.3.1.9.1 In some circumstances it is possible for the variable value to be in neither the upper zone nor the lower zone. This may occur upon definition of a new limit or upon equipment startup when the value of the variable lies in the deadband. In this case, the active state of the limit is considered to be NO ZONE. The limit shall remain in this state until the variable value reaches either boundary of the deadband.

7.3.4.3.2 *Modification of Limit Values* — Values for the monitoring limits on any monitored variable may be modified by the host using the message transaction S2,F45/F46 (Define Variable Limit Attributes). The equipment must reject any S2,F45 message which contains limit information which conflicts with the following rules:

- LIMITMAX≥UPPERDB≥LOWERDB≥LIMIT-MIN;
- If either UPPERDB or LOWERDB is defined, both must be defined; if either UPPERDB or LOWERDB is undefined, both must be undefined.

7.3.4.3.2.1 The first rule is defined and graphically depicted in Figure 8. The second rule refers to the host's ability to turn any limit 'on' or 'off'. While a minimum of seven limits must be available for each monitored variable, it will be common for the host application to require less than seven or even none of the limits be used. The limits not needed can be disabled by leaving the values for UPPERDB and LOWERDB 'undefined'. Limits may be disabled for a VID or for all monitored VIDs by using zero length lists in the S2,F45 message.

7.3.4.3.2.2 All monitored variables must be one of three types: integer, floating point, or Boolean. This may be accomplished by using the following formats: 11, 20, 3(), 4(), 5().

NOTE 6: The binary format is not allowed. If the ASCII format is used, the equipment shall perform a conversion into one of the numeric types before performing any value comparisons, both for limit validations and zone transitions.

7.3.4.3.3 *Limit Values Request* — The host may request the current limit values for a specified VID using the message transaction S2,F47/F48 (Variable Limit Attribute Request).

7.3.4.4 *Requirements*

- A minimum of seven limits per monitored variable must be available.
- One CEID per monitored variable must be supplied for zone transition reporting.
- All limit definitions must be kept in nonvolatile storage.
- The equipment must enforce the limit validation rules defined above.
- The specification and documentation of which variables may be monitored with this capability is the responsibility of the equipment manufacturer based on the specific instance of equipment. This subject also may be addressed by equipment models of classes of semiconductor equipment.

**Table 4 Limit State Transition Table**

| # | *Current State* | *Trigger* | *New State* | *Action* | *Comment* |
|---|---|---|---|---|---|
| 1 | DISABLED | Limit attributed defined w/ S2,F45. | ENABLED | None | The substate of ENABLED is determined by the current value of the monitored variable. |
| 2 | ENABLED | Limit attributes set to undefined w/ S2,F45. | DISABLED | None | None |
| 3 | BELOW LIMIT | Variable Increased to be ≥UPPERDB | ABOVE LIMIT | None | Zone Transition. |
| 4 | ABOVE LIMIT | Variable decreases to be ≤LOWERDB | BELOW LIMIT | None | Zone Transition. |
| 5 | NO ZONE | Variable decreases to be ≤LOWERDB | BELOW LIMIT | None | Zone Transition. |
| 6 | NO ZONE | Variable increases to be ≥UPPERDB | ABOVE LIMIT | None | Zone Transition. |

LETTER BALLOT

7.3.4.5 *Scenarios*

7.3.4.5.1 Zone Transition Event occurs in equipment:

| COMMENTS | HOST | EQUIPMENT | COMMENTS |
|---|---|---|---|
| Send enabled event report to host as shown in § 5.3.1. | | | |

7.3.4.5.2 Host defines Limit Attributes:

| COMMENTS | HOST | EQUIPMENT | COMMENTS |
|---|---|---|---|
| [IF] S2,F45 is Multi-block | | | |
| [THEN] Send Multi-block inquire | S2,F39--> | | |
| | | <--S2,F40 | Multi-block grant. |
| [END IF] | | | |
| Host defines new variable limit attributes. | S2,F45--> | | |
| | | <--S2,F46 | Equipment acknowledges host request. |

7.3.4.5.3 Host queries equipment for current Limits:

| COMMENTS | HOST | EQUIPMENT | COMMENTS |
|---|---|---|---|
| Host queries equipment for current variable limit attribute definitions. | S2,F47--> | | |
| | | <--S2,F48 | Equipment returns report containing requested variable limit attribute values. |

7.3.5 *Status Data Collection*

7.3.5.1 *Purpose* — This capability allows the host to query the equipment for selected status information and is useful in synchronizing with equipment status.

7.3.5.2 *Definitions*

7.3.5.2.1 *Status variable value (SV)* — A data item containing the value of a status variable. See SEMI E5 for a full definition of this data item.

7.3.5.2.2 *Status variable ID (SVID)* — A unique identifier of a status variable. See SEMI E5 for a full definition of this data item.

7.3.5.3 *Detailed Description* — The host may query equipment status by specifying the desired SVID's. Upon such a request, the equipment sends the host the value of the selected status variables. The host also may request the description (name and units) of any or all available status variables.

7.3.5.4 *Requirements*

- The equipment manufacturer must provide unique SVID's for the various status variables (SV) available for data collection in the equipment.
- All status data is available for status data collection. See § 6.4, Variable Item List, for a list of status variables.
- All SV's must contain valid data whenever provided to the host.

LETTER BALLOT

7.3.5.5 *Scenarios*

7.3.5.5.1 Request Equipment Status Report:

| COMMENT | HOST | EQUIPMENT | COMMENT |
|---|---|---|---|
| Host requests report of selected status variable values | S1,F3--> | | |
| | | <--S1,F4 | Equipment responds with the requested status variable data. |

7.3.5.5.2 Request Equipment Status Variable Name list:

| COMMENT | HOST | EQUIPMENT | COMMENT |
|---|---|---|---|
| Host requests equipment to identify selected status variables. | S1,F11--> | | |
| | | <--S1,F12 | Equipment responds with the requested status variable descriptions. |

7.3.6 *On-Line Identification*

7.3.6.1 *Purpose* — Implementation of SEMI E5 (a GEM Fundamental Requirement) requires the equipment to accept the S1,F1 from the Host at any time while it is ONLINE and COMMUNICATING, and respond with S1,F2. The On-line Identification capability describes the host-initiated scenario. The equipment-initiated scenario is used for a different purpose and is defined in § 4.5 and § 5.13 describing the GEM 'Control Model'.

7.3.6.2 *Definitions*

7.3.6.2.1 *Equipment Model Type (MDLN)* — ASCII string containing the equipment model. See SEMI E5 for a full definition of this data item.

7.3.6.2.2 *Equipment Software Revision Code (SOFTREV)* — ASCII string containing the equipment software revision. See SEMI E5 for a full definition of this data item.

7.3.6.3 *Detailed Description* — On-line Identification allows the host to verify the presence and identity of the equipment.

7.3.6.4 *Requirements* — (Host-Initiated) an S1,F2 response from the equipment must provide MDLN and SOFTREV information which reflects the hardware and software configuration of the equipment.

7.3.6.4.1 SOFTREV must uniquely identify different releases of equipment software. Any change in equipment software must result in a corresponding change to SOFTREV.

7.3.6.4.2 The equipment-initiated S1,F1 is not required except as described in § 4.5 and § 5.13 describing the GEM 'Control Model'.

7.3.6.5 *Scenario*

7.3.6.5.1 Host Initiated:

| COMMENT | HOST | EQUIPMENT | COMMENT |
|---|---|---|---|
| Are you there? | S1,F1--> | | |
| | | <--S1,F2 | Equipment replies with MDLN and SOFTREV. |

7.4 *Alarm Management* — The alarm management capability provides for host notification and management of alarm conditions occurring on the equipment.

7.4.1 *Purpose* — Historically, a precise definition of an equipment alarm has been absent. Consequently, differing interpretations have resulted in inconsistent implementations. This is addressed by providing a more rigorous definition (see definition in § 5.4.2 below) of an alarm.

7.4.1.1 In addition, it is often important for equipment to report more extensive information to the host than has been available in the S5,F1/F2 (Alarm Report Send/Acknowledge) transaction. The data required in such cases is very dependent on equipment type, host information requirements, and alarm situation. This issue is addressed by providing event reporting methods that are tied to alarm state changes.

- Lastly, the alarm management capability provides mechanisms for:
- Reporting the time of an alarm state change,
- Uploading a list of alarm texts,
- Enabling and disabling the notification of specific alarms, and
- Host query of alarms set and enabled status on the equipment.

7.4.2 *Definitions*

7.4.2.1 *Alarm* — An alarm is related to any abnormal situation on the equipment that may endanger people, equipment, or material being processed. Such abnormal situations are defined by the equipment manufacturer based on physical safety limitations. Equipment activities potentially impacted by the presence of an alarm shall be inhibited.

7.4.2.1.1 Note that exceeding control limits associated with process tolerance does not constitute an alarm nor do normal equipment events such as the start or completion of processing.

7.4.2.2 *AlarmsEnabled* — Status value consisting of a list of the alarm ID's currently enabled for reporting to the host. See SEMI E5 for a full definition of this variable data item.

7.4.2.3 *AlarmsSet* — Status value consisting of a list of the alarm ID's currently in the ALARM SET (or unsafe) state. See SEMI E5 for a full definition of this variable data item.

7.4.2.4 *ALCD* — Alarm code data item used in the S5,F1 (Alarm Report Send) and S5,F6 (List Alarm Data) messages. This code is divided into two parts, the alarm set/cleared bit and the 7 bit alarm category code. Only the set/cleared bit is used—bit 8 = 1 means alarm set, = 0 means alarm cleared. The alarm category code is not used. See SEMI E5 for a full definition of this data item.

7.4.2.5 *ALID* — Alarm identifier. See SEMI E5 for a full definition of this data item.

7.4.2.6 *ALTX* — Data item contained in the S5,F1 and S5,F6 messages containing a brief textual description of an alarm. See SEMI E5 for a full definition of this data item.

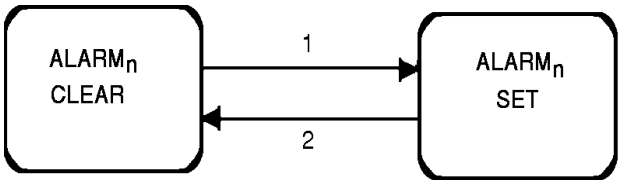



**Figure 11**
**State Diagram for Alarm ALIDn**

7.4.3 *Detailed Description* — Two alarm notification mechanisms are defined to achieve the flexibility necessary for the reporting required by host systems. First, stream 5 alarm reporting enables a brief, yet fixed, method for notification of alarm occurrences using the S5,F1/F2 transaction. Second, to address the host's potential need for more extensive and flexible data reporting, two collection events ('alarm set' and 'alarm cleared') are defined for each possible alarm

condition on the equipment to allow the use of event data collection mechanisms. In the latter case, reports are sent by the equipment using the Event Report/Acknowledge transaction (see § 5.3 on event data collection).

7.4.3.1 The $alarm_n$ detected and cleared events are derived from the state model for an alarm (see Figure 10 and Table 5). In this model each of n alarms can be in one of two possible states, either ALARM CLEAR or ALARM SET. The transition from the ALARM CLEAR to the ALARM SET state is defined as the collection event '$Alarm_n$ detected' (transition 1). Conversely, the transition from ALARM SET to ALARM CLEAR is defined as the collection event '$Alarm_n$ cleared' (transition 2).

NOTE 7: The alarm capability is intended as an addition to standard safety alarms (e.g., lights, horns). There is no intent to replace direct operator notification of such problems, nor is there the expectation that the host can prevent or directly address such alarms.

**Table 5 Alarm State Transition Table**

| # | *Current* | *Trigger* | *New State* | *Action* | *Comment* |
|---|---|---|---|---|---|
| 1 | $ALARM_n$ CLEAR | $Alarm_n$ is detected on the equipment. | $ALARM_n$ SET | Initiate local actions (if any) to ensure safety. Update AlarmsSet and $ALCD_n$ values. Generate and issue alarm message if enabled. | Inhibited activities require operator or host intervention prior to resuming. |
| 2 | $ALARM_n$ SET | $Alarm_n$ is no longer detected on the equipment. | $ALARM_n$ CLEAR | Update AlarmsSet and $ALCD_n$ values. Generate and issue alarm message if enabled. | Inhibited activities require operator or host intervention prior to resuming. |

7.4.3.2 The equipment manufacturer is responsible for identifying all alarms on their equipment by:

- Applying the above alarm definition,
- Consulting Related Information § R1-3 ~~Application Note § A.3~~ for examples of alarms for various equipment configurations,
- Noting that the presence of an alarm typically requires some action or intervention before resuming safe operation of the equipment, and by
- Referring to Table 6 below which delineates the differences between events and alarms.

**Table 6**

| *EVENT* | *ALARM* |
|---|---|
| Any occurrence detectable by the equipment. | Related to only those occurrences that are abnormal, undesirable, AND endanger people, equipment, or physical material being processed. |
| Certain events may trigger a state transition(s). | Each alarm has an associated two-state state model: ALARM SET (or unsafe) and ALARM CLEAR (or safe). |
| Equipment activities are not necessarily inhibited by the occurrence of an event (unless it is associated with an alarm or intentional inhibit). | The presence of an alarm inhibits equipment activities to ensure safe operation until the alarm condition is cleared. |
| Certain events may occur in an expected sequence. | Alarms may occur at any time. |

7.4.3.3 *Enable/Disable Alarms* — Upon request from the host, the equipment shall enable or disable reporting of certain alarms. Enabling or disabling a given alarm shall impact the communication of both the alarm set and clear messages equally (i.e., turn them both on or both off). This is not the case for enabling/disabling of the associated collection events, where the alarm-set and alarm-cleared events can be enabled and disabled separately. The current enable/disable settings must be stored in nonvolatile memory. Changes to the enable/disable settings must be reflected in the AlarmsEnabled status value.

NOTE 8: The alarm itself is not being enabled or disabled, but the reporting of the alarm through SECS-II messages is being enabled or disabled.

7.4.3.4 *Send Alarm Report* — Upon detecting a change in the status of a given alarm, an associated alarm state model shall transition to the opposite state. Following initiation of local actions necessary to ensure safety, the equipment must update the AlarmsSet and associated ALCD values and send an alarm message and/or an event message to the host assuming one or both are enabled. Alarm messages must be sent before the corresponding event messages if both are enabled.

NOTE 9: The ALCD is divided into two parts, the alarm set/cleared bit and the 7 bit alarm category code. Only the set/cleared bit is used—bit 8 = 1 means alarm set, = 0 means alarm cleared. The alarm category code is not used.

7.4.3.5 *List Alarm Text* — Upon request from the host, the equipment sends values of alarm text associated with a specified list of alarm ID's using the S5,F5/F6 (List Alarms Request/Data) transaction.

7.4.3.6 *List Currently Set Alarms* — The host may obtain a listing of currently set alarms by employing any data collection method specified by GEM and referencing the variable data item called 'AlarmsSet' (e.g., including " AlarmsSet in a report definition). When reported, AlarmsSet must contain a list of all currently set alarms. Each alarm set or clear occurrence must cause a change to the AlarmsSet status value prior to reporting it to the host.

7.4.3.7 *List Currently Enabled Alarms* — The host may obtain a listing of currently enabled alarms by employing any data collection method specified by GEM and referencing the variable data item called 'AlarmsEnabled' (e.g., including AlarmsEnabled in a report definition). When reported, AlarmsEnabled must contain a list of all alarm ID's currently enabled for reporting. The equipment must update the value of AlarmsEnabled upon corresponding change(s) to the enable/disable settings.

7.4.4 *Requirements*

- A set of alarms relating to the physical safety limits of operator, equipment, or material being processed must be defined for the equipment by the equipment manufacturer.
- The equipment must maintain all enable/disable states and report definitions for alarms and collection events in nonvolatile memory.
- Each alarm defined on the equipment must have a brief description of its meaning, an associated unique alarm identifier (ALID), alarm text (ALTX), an alarm status (ALCD) and two unique collection event identifiers (CEIDs), one for set and one for cleared.
- Enabled alarm reports must be sent prior to corresponding enabled event reports.

7.4.5 *Scenarios*

NOTE 10: Consult event reporting sections of this Document for descriptions of enabling, disabling, and sending collection event reports.

7.4.5.1 Enable/Disable Alarms:

```
COMMENTS                        HOST    EQUIPMENT    COMMENT

Enable/disable Alarm            S5,F3-->
                                        <--S5,F4     Acknowledge
```

7.4.5.2 Upload Alarm Information:

```
COMMENTS                        HOST    EQUIPMENT    COMMENT

Request alarm data/text         S5,F5-->
                                        <--S5,F6     Send alarm data/text
```

LETTER BALLOT

7.4.5.3 Send Alarm Report:

| COMMENTS | HOST | EQUIPMENT | COMMENT |
|---|---|---|---|
| Alarm occurrence detected by the equipment. | | | |
| | | <--S5,F1 | Send alarm report (if enabled). |
| Acknowledge | S5,F2--> | | |
| | | <--S6,F11 | Send event report (if enabled). |
| Acknowledge | S6,F12--> | | |

7.5 *Remote Control*

7.5.1 *Purpose* — This capability provides the host with a level of control over equipment operations.

7.5.2 *Definitions*

7.5.2.1 *Host Command Parameter (CPNAME/CPVAL/CEPVAL)* — A parameter name/value associated with a particular host command (S2,F41/S2,F49). The equipment manufacturer must provide unique names (CPNAMEs) for any supported command parameters. Command parameters are not specified in this Document but are left to equipment manufacturers to define. Equipment models of specific classes of semiconductor equipment also may address this issue. Note that, if there are no associated parameters a zero-length list is sent. The data item CEPVAL, which can be defined as a list, allows grouping of related parameters within a main parameter. If the CEPVAL is defined as a single (non-list) item, then it is the equivalent of a CPVAL.

7.5.2.1.1 The uses of OBJSPEC in the header structure of the S2,F49 Enhanced Remote Command allows the equipment supplier to define a set of unique identifiers for different objects within the equipment such as: equipment subsystems, subsystem components, processing stations, ports, and exchange stations.

7.5.3 *Description* — The equipment responds to host commands that provide the following functions relative to individual equipment implementations:

- Start processing
- Select a process program or recipe
- Stop processing
- Temporarily suspend processing
- Resume processing
- Abort processing

7.5.3.1 Additional commands may be implemented by the equipment manufacturer (e.g., vent chamber, clear material, open door).

7.5.3.2 Remote commands shall be interpreted as ‘request action be initiated’ rather than ‘do action’. The equipment may then respond via S2,F42/S2,F50 with HCACK = 4 if the command ‘is going to be performed’. This alleviates any transaction timeouts for commands that may take a long time to perform. The completion of the action initiated by the remote command (i.e., HCACK = 0 or 4) must result in either a state transition or other action that generates a collection event upon normal/abnormal completion.

7.5.3.3 The format for all remote commands is ASCII, with a maximum length of 20 characters. The character set is restricted to the printable characters (hexadecimal 21–7E). Note that spaces are not allowed.

7.5.3.4 The following remote commands (RCMDs), if implemented on the equipment, shall be supported as described below (see § 4.6 for a description of Equipment Processing States).

NOTE 11: The terms ‘current cycle’ and ‘safe break point’ used below are to be defined by the supplier or within the models of classes of semiconductor equipment.

7.5.3.5 *START* — This command is available to the host when a process program or recipe has been selected and the equipment is in the 'ready' processing state. The START command instructs the equipment to initiate processing. Variable parameter settings may be included as name/value command parameters CPNAME/CPVAL/CEPVAL.

7.5.3.6 *PP-SELECT* — This command instructs the equipment to make the requested process program (s) available in the execution area. The process programs (PPIDs) are specified via the command parameter list. A status variable (PPExecName) contains the PPID of the process program(s) currently selected. PP-SELECT and PPExecName shall be implemented if process program is the Process Recipe type implemented.

7.5.3.7 *RCP-SELECT* — This command uses the Enhanced Remote Command S2,F49 to instruct the equipment to prepare the requested recipes for execution in the execution area. The recipes and variable parameters are specified via command parameter lists. Each recipe specification may be accompanied by new variable parameter settings, if any, in the command parameter list. A status variable RcpExecName contains the recipe specifiers or identifiers of the recipes currently selected. RCP-SELECT and RcpExecName shall be implemented if the equipment has implemented E42 recipes or E139 recipes.

7.5.3.8 *STOP* — Command to complete the current cycle, stop in a safe condition and return to the 'idle' processing state. Stop has the intent of stopping the process. The equipment is not required to support the continuation of processing. Stop leaves material either fully processed or partially processed so that the processing can be later completed. For example, for a single wafer process tool, five wafers have been processed while the remaining wafers remain unprocessed.

7.5.3.9 *PAUSE* — Command to suspend processing temporarily at the next safe break point. Pause has the intent of resuming the process at the same point where it was paused. The process may be RESUMED, STOPPED, or ABORTED while in a PAUSED condition. RESUME shall be able to continue the process from the same point where it was paused.

7.5.3.10 *RESUME* — Command to resume processing from the point where the process was paused.

7.5.3.11 *ABORT* — Command to terminate the current cycle prior to its completion. Abort has the intent of immediately stopping the process and is used because of abnormal conditions. Abort makes no guarantee about the subsequent condition of material. In the above example, the wafers being processed at the time of the abort may not be completely processed. Other AbortLevels >1 may be defined by the manufacturer or addressed by models of specific classes of semiconductor equipment.

7.5.3.12 CPNAME = AbortLevel, CPVAL = 1 means terminate current cycle at the next 'safe break point', retrieve all material, stop in a safe condition and return to the idle state in the processing state machine.

7.5.4 *Requirements*

- The following Remote Commands, as defined under Descriptions, must be implemented on equipment to satisfy minimum requirements for this capability:
  - START
  - STOP
- The RCMD value for all commands supported on the equipment must be recognized if sent with all uppercase characters (e.g., 'STOP', 'START', 'PP-SELECT', 'PAUSE', etc.). In addition to accepting strings with all uppercase characters, the equipment can optionally accept strings with all lowercase characters or mixed-case strings. The equipment documentation should describe whether or not the optional lowercase or mixed-case strings are supported.
- Stream 2 currently provides for Host Command Send and Enhanced Remote Command. The equipment shall support one or both methods, based on appropriateness.
- The Enhanced Remote Command is used to address size, complexity, or the need to target a specific subsystem within the equipment, (i.e., processing station, port, exchange station, material handler, chamber).

LETTER BALLOT

7.5.5 *Scenarios*

7.5.5.1 Host Command Send Scenario:

| COMMENTS | HOST | EQUIPMENT | COMMENT |
|---|---|---|---|
| Host Command | Send S2,F41--> | | |
| | | <--S2,F42 | Host Command Acknowledge |
| | | | [IF] Command Accepted (HCACK = 0 or 4) |
| | | <--S6,F11 | [THEN] Event Report-state change or other collection event occurrence. |
| Event Report Acknowledge | S6,F12--> | | |

7.5.5.2 Enhanced Remote Command Scenario:

| COMMENTS | HOST | EQUIPMENT | COMMENT |
|---|---|---|---|
| Enhanced Remote Command | S2,F49--> | | |
| | | <--S2,F50 | EnhancedRemote Command Acknowledge |
| | | | [IF] Command Accepted (HCACK = 0 or 4) |
| | | <--S6,F11 | [THEN] Event Report-state change or other collection event occurrence. |
| Event Report Acknowledge | S6,F12--> | | |

7.6 *Equipment Constants*

7.6.1 *Purpose* — This capability provides a method for the host to read and to change the value of selected equipment constants on the equipment.

7.6.2 *D*efinitions — None.

7.6.3 *Description* — This capability allows the host to reconfigure equipment constants to support a variety of situations. The following functions are included:

7.6.3.1 *Host Sends Equipment Constants* — Allows the host to change the value of one or more equipment constants.

7.6.3.2 *Host Equipment Constant Request* — Allows the host to determine the current value of equipment constants.

7.6.3.3 *Host Equipment Constant Namelist Request* — Allows the host to retrieve basic information about the equipment constants available at the equipment.

7.6.4 *Requirements*

- Equipment constants must be stored in nonvolatile memory.
- The equipment must be in a ‘safe’ condition to accept new constant(s) settings as defined by the equipment manufacturer.
- The equipment must provide a collection event to alert the host whenever an equipment constant is changed by the operator. Information indicating which constant was changed shall be available for the event report.

LETTER BALLOT

7.6.5 *Scenarios*

7.6.5.1 Host Sends Equipment Constants:

| COMMENTS | HOST | EQUIPMENT | COMMENTS |
|---|---|---|---|
| Host sends equipment constants | S2,F15--> | | |
| | | <--S2,F16 | EAC = 0 equipment sets constants |

7.6.5.2 Host Equipment Constants Request:

| COMMENTS | HOST | EQUIPMENT | COMMENTS |
|---|---|---|---|
| Host constant request | S2,F13--> | | |
| | | <--S2,F14 | Equipment constant data |

NOTE 12: This capability also can be accomplished using S6,F19 and S6,F20. See § 5.3.

7.6.5.3 Host Equipment Constant Namelist Request:

| COMMENTS | HOST | EQUIPMENT | COMMENTS |
|---|---|---|---|
| Host constant namelist request | S2,F29--> | | |
| | | <--S2,F30 | Equipment constant namelist |

7.6.5.4 Operator Changes Equipment Constant:

| COMMENTS | HOST | EQUIPMENT | COMMENTS |
|---|---|---|---|
| | | | Operator changes equipment constant at equipment operator console. |
| | | <--S6,F11 | Equipment reports equipment constant change. |
| Host acknowledges event | S6,F12--> | | |

7.7 *Process Recipe Management* — Specifications for equipment processing (e.g., recipes) must be managed through interaction between the equipment and the host system.

7.7.1 *Purpose* — Process Recipe Management provides a means to transfer process specifications and to share the management of these specifications between the host and the equipment.

7.7.1.1 *Meeting the Purpose* — An equipment shall provide compliant Process Recipe Management by implementing one of the following:

7.7.1.1.1 *Process Programs* — The details on how to implement process programs are found in the remainder of § 5.7, beginning at ¶ 5.7.2.

7.7.1.1.2 *E42 Recipes* — The details on how to implement recipes are found in the remainder of § 5.7, beginning at ¶ 5.7.2.

7.7.1.1.3 *E139 Recipes* — E139 recipes are defined by SEMI E139 Recipe and Parameter Management (RaP). Some of the requirements for Process Recipe Management in this Standard (SEMI E30) apply to equipment that has chosen to implement RaP, but many do not (as they apply specifically to process programs and/or E42 recipes).

7.7.1.1.4 Where applicable, Process Recipe Management requirements which are specific to process programs, E42 recipes, or E139 recipes are labeled as such.

7.7.2 *Definitions*

7.7.2.1 *Process Recipe* — General term that refers to any executable specification of a process or activity on the equipment. Process programs, E42 recipes, and E139 recipes are specific types of Process Recipes.

7.7.2.2 *Process Program* — Specific type of Process Recipe as defined in this Document. See further definition of process program and related definitions in § 5.7.2.1.

7.7.2.3 *E139 Recipe* — Specific type of Process Recipe as defined in SEMI E139. An E139 recipe is a collection of one or more recipe components or Process Definition Elements (PDE's). See SEMI E139 for further detail.

7.7.2.4 *E42 Recipe* — Specific type of Process Recipe as defined in SEMI E42. See related definitions in § 5.7.2.2. For further detail, see SEMI E42.

7.7.2.5 *PPError* — A text data value with information about verification errors of a process program that failed verification. If the equipment provides an event for recipe verification and/or recipe verification failure, then PPError shall be a DVVAL. Otherwise, PPError shall be an SV. PPError is not required for equipment implementing E139 recipes.

7.7.2.6 *PPFormat* — A variable (SV) indicating the type or types of process programs and recipes that are supported. PPFormat is not required for equipment implementing E139 recipes for Process Recipe management.

1 = Unformatted process programs

2 = Formatted process programs

3 = Both unformatted and formatted process programs

4 = Execution recipes

5 = Large unformatted process programs

6 = Large formatted process programs

7 = Both large unformatted and large formatted process programs

8 = Large execution recipes

and a combination of these formats. See SEMI E5 for a complete list.

7.7.2.7 *Definitions for Process Programs*

7.7.2.7.1 *Process Program* — A process program is the pre-planned and reusable portion of the set of instructions, settings, and parameters under control of the equipment that determine the processing environment seen by the manufactured object and that may be subject to change between runs or processing cycles.

7.7.2.7.2 *Process Program Identifier* — A text string (PPID) used to identify a process program.

7.7.2.7.3 *Formatted Process Program* — A process program that is presented as an ordered sequence of command codes with their associated parameters as dictated by S7,F23, and S7,F26. Where formatted process programs are supported, equipment must also provide information sufficient to allow a user at the host to create, display, modify, and partially verify their contents (e.g., that information provided in S7,F22).

7.7.2.7.4 *Unformatted Process Program* — An unformatted process program is transferred without structure as the single data item PPBODY (refer to SEMI E5 for a complete description of PPBODY).

7.7.2.7.5 *Process Program Change Event* — The collection event associated with the occurrence of the creation, modification, or deletion of a process program by the operator.

7.7.2.7.6 *PPChangeName* — A data value (DVVAL) containing the PPID of the process program affected by the event Process Program Change Event. See SEMI E5 for a full definition of this variable data item.

7.7.2.7.7 *PPChangeStatus* — The action taken on the process program named by PPChangeName. This variable is valid for the collection event Process Program Change Event. See SEMI E5 for a full definition of this variable data item.

7.7.2.7.8 *PPExecName* — The status variable containing the PPID (s) of the currently selected process program (s). See SEMI E5 for a full definition of this variable data item.

7.7.2.7.9 *PP-SELECT* — The remote command used to select one or more process programs for execution. The process programs are specified by PPID via the command parameter list.

7.7.2.7.10 *Process Program Verification* — Verification is syntax checking of a process program. Verification ensures only that a process program is structured correctly. It does not ensure that the program has the correct parameters to run a particular process or product (see Process Program Validation). Equipment supporting unformatted process programs should provide a variable DVVAL PPError that provides information to the user concerning the error or errors when an attempt to verify a process program fails.

NOTE 13: It may not be possible for the equipment to verify unformatted process programs other than to check the size of the program and internal program checksums. Equipment has no standard means of indicating the type of error encountered in an unformatted process program.

7.7.2.7.11 *Process Program Validation* — Validation is type-and-range checking of parameters in a process program, and is performed after verification.

7.7.2.8 *Definitions for Recipes*

7.7.2.8.1 *Execution recipe* — A type of recipe stored by the equipment for purposes of editing, verification, and execution.

7.7.2.8.2 For complete definitions of execution recipes and their standard attributes, see SEMI E42, § 6.

7.7.2.8.3 *Execution Recipe Change Event* — The collection event associated with the occurrence of the modification or deletion of an execution recipe stored by the equipment.

NOTE 14: A recipe is modified whenever its body is changed.

7.7.2.8.4 *New Execution Recipe Event* — The collection event associated with the creation of a new execution recipe at the equipment.

7.7.2.8.5 *Object form recipe* — A recipe with body in a proprietary format that may be presented without structure.

7.7.2.8.6 *RcpChangeName* — A data value (DVVAL) containing the identifier of the recipe affected by the event Execution Recipe Change Event or New Execution Recipe Event. See the SEMI E5 Standard for a full definition of this variable data item.

7.7.2.8.7 *RcpChangeStatus* — The action taken on the recipe named by RcpChangeName. This variable is valid for the collection event Execution Recipe Change Event or New Execution Recipe Event. See the SEMI E5 Standard for a full definition of this variable data item.

7.7.2.8.8 *RCP-SELECT* — The remote command used to select one or more recipes for execution. See § 5.5.3.

7.7.2.8.9 *Recipe Attribute* — Information about the recipe that is transferred with the recipe as a name/value pair. The value may be a single item or a list.

7.7.2.8.10 *Recipe* — A recipe contains both a set of instructions, settings, and parameters that the equipment uses to determine the processing environment (its body or process program) and a set of attributes that provide information about the recipe, such as the date and time the body was last changed.

7.7.2.8.11 SEMI E42 defines two types of recipes: *managed recipes* and *execution recipes*. For purposes of GEM, the term *recipe* refers to an *execution recipe* only.

7.7.2.8.12 *Recipe Identifier* — A recipe identifier is a formatted text string (RCPID) used to identify the recipe.

7.7.2.8.13 *Recipe Specifier* — A formatted text string (RCPSPEC) used in messages to indicate a specific recipe. A recipe specifier includes the recipe identifier. It may also include additional information, such as the name of the specific component of the equipment where the recipe is to be executed (e.g., a process chamber) and the name of a recipe repository on the host.

7.7.2.8.14 *Recipe Verification* — Verification is syntax checking of a recipe's body. Verification ensures that a recipe body is structured correctly and has the correct syntax. It may also provide a check of semantics. It does not ensure that the body has the correct parameters to run a particular process or product (see Recipe Validation).

NOTE 15: Unverified recipes shall be verified upon download.

7.7.2.8.15 *Recipe Validation* — Validation is type-and-range checking of parameters in a recipe, and is performed when the recipe is selected for execution. The recipe may be correct in its syntax and semantics but should fail validation if it cannot be executed with the current equipment configuration.

7.7.2.8.16 *Source Form Recipe* — A recipe with a body that is presented as an ordered sequence of text. A source form recipe may be created and edited off-line to the equipment. Definition of syntax requirements shall be documented, in order to allow proper off-line editing.

7.7.2.8.17 *Variable Parameters* — Variable parameters are recipe parameters that are defined in the body of the recipe and whose run-time values may be set outside of the recipe when the recipe is selected for execution and/or when processing is started. Both the host and the operator may specify new settings as a parameter name/value pair.

7.7.2.8.18 *Variable Parameter Definition* — A variable parameter definition has three parts: the name of the variable parameter, its default setting, and restrictions on the run-time value selected. Variable parameter definitions are stored in the recipe attribute 'Parameters'.

7.7.2.9 *Definitions Applicable to E42 Recipes and E139 Recipes*

7.7.2.9.1 *RcpExecName* — The status variable containing the specifiers of the currently selected recipe(s). See the SEMI E5 Standard for a full definition of this variable data item.

7.7.3 *Description*

7.7.3.1 *Process Program Description*

7.7.3.1.1 Process programs allow the equipment's process, and/or the parameters used by that process, to be set and modified by the engineer to achieve different results. Different process programs may be required for different products, while often the same process program will be used for all lots of a given product. The engineer must be able to create such programs, to modify current programs, and to delete programs from equipment storage.

7.7.3.1.2 For the host to ensure that the proper process programs are in place at the equipment, there must be a means of transferring them from equipment to host and from host to equipment. The host also may need to delete process programs from the equipment's storage to make room for a process program to be downloaded. In addition, the host must be kept informed whenever a local change occurs in the contents or status of a process program.

7.7.3.1.3 Both formatted and unformatted process programs may be uploaded and downloaded. This capability provides for both host- and equipment-initiated transfers. The equipment-initiated transfer may be used at the request of the process engineer or operator at the equipment.

7.7.3.1.4 If a process program exists with the same PPID as the one given in the SECS-II message, the old process program must be replaced. The PPID in the e process program in nonvolatile storage.

7.7.3.2 *E42 Recipe Description*

7.7.3.2.1 Specifications in § 5.7.3.1 apply to E42 recipes as well as process programs, with the following differences:

- A recipe contains a body corresponding to a process program. In addition, it contains attributes defined for execution recipes in SEMI E42, § 6. Recipe attributes are transferred whenever the recipe is downloaded or uploaded.
- The same SECS-II messages are used for all execution recipes, regardless of the internal structure of the recipe body.
- If an execution recipe already exists with the same identifier as the one given in the SECS-II message, the downloaded recipe shall be rejected (not stored) unless the host has specified a 'forced overwrite' in the data item RCPOWCODE.

- A recipe currently being edited shall be protected from inadvertent change or overwriting by a recipe with the same identifier that is downloaded during this time. If the downloaded recipe is accepted (stored), the equipment shall require the operator either to save the edited recipe to a new (unused) identifier or to discard it.
- For the equipment to initiate either an upload or download of a recipe, it shall request the host to initiate an upload or download procedure. In addition, it may be necessary to also specify the name of the repository (recipe namespace) at the host.

7.7.3.3 *Large Process Programs and E42 Recipes*

7.7.3.3.1 Process programs and recipes for certain types of equipment, such as metrology or inspection equipment, contain images and are, therefore, very large. A process program or recipe cannot be sent as a single item in a SECS-II message if its size exceeds SECS-II data item size limits as defined in SEMI E5, or if the total message size would exceed the SECS message size limits defined in transport protocol Standards such as SEMI E4 and SEMI E37. Furthermore, process programs and recipes may also require some preparation prior to transfer.

7.7.3.3.2 The commonly used Stream 7 and Stream 15 transfer functions require that the entire process program be sent in a single message. Thus, it is not possible to send a large process program or recipe with such messages. However, there is an alternative set of Stream 7 and Stream 15 functions that supports large transfers. These messages invoke Stream 13 Data Set Transfer Protocol messages to transfer large process programs (or recipes) by using a sequence of read messages. The completion of such a read transaction is indicated by 'ERROR: End of Data'. The Data Set Transfer Protocol does not set any limit on the size of the data set.

7.7.3.4 *E139 Recipe Description*

7.7.3.4.1 As an alternative to process programs or E42 recipes as defined above, equipment may implement recipes according to SEMI E139, Recipe and Parameter Management (RaP). SEMI E139 defines communications for recipe management among the factory system, equipment, and recipe editors. It also defines certain user-accessible, descriptive information that is included with the recipe to facilitate recipe management. E139 recipes support the following features:

- *On-Tool & Off-Tool Recipe Management* — Support the management of multi-part recipes within the equipment as well as at the FICS level without the requirement that these management applications be integrated.
- *Recipe Integrity* — Guarantee that the recipes that define process execution on a piece of equipment are the same as those intended for that use within the FICS.
- *Process Integrity* — Ensure that all configuration values/settings of the tool that can be changed by the user and which affect the process outcome can be managed by the host within the context of a process job.
- *Adjustable Parameter Definition* — Support the definition of recipe parameters to be used by other SEMI Standards (e.g., SEMI E40) to adjust the parameter values in a consistent manner across all equipment.
- *On-tool & Off-Tool Recipe Creation & Editing* — Define a consistent interaction protocol between the FICS and the systems(s) that support recipe creation/editing functionality.
- *Information Accessibility* — Make key information about the recipes visible to the FICS.

7.7.4 *Requirements*

- The equipment manufacturer shall provide a method to create, modify, and delete Process Recipes. This method shall exist on either the equipment or on a separate computing system.
- If the equipment manufacturer has implemented process programs, a CEID shall be defined for a collection event for the creation, the deletion, or the modification (completion of an editing session) of a process program (Process Program Change Event).
- If the equipment manufacturer has implemented E42 recipes, there are two separate collection events with corresponding CEIDs, one for the creation of a new recipe (New Execution Recipe Event) and one when a recipe is changed or deleted (Execution Recipe Change Event). A New Execution Recipe Event shall occur whenever a new recipe identifier is created through download, edit, copy, or rename operations. An Execution Recipe Change Event shall occur whenever the body of an existing recipe is modified.

LETTER BALLOT

- If the equipment manufacturer has implemented process programs or E42 recipes, the name (identifier) that the engineer or operator uses to refer to the process program or recipe is the same as the identifier used by the host.
- Upon request from the host or operator, the equipment shall perform the following actions with regard to Process Recipes stored in nonvolatile storage: upload, download, delete, and list current equipment Process Recipe directory.
- The equipment shall be able to store in nonvolatile memory the number of Process Recipes sufficient to execute three unique process cycles. For example, if a wire-bonder requires both an 'ALIGN' process program and a 'BOND' process program for a full process cycle, then it must provide nonvolatile storage for at least three pairs of process programs. These stored Process Recipes may not be modified in any way by the execution process, nor may the execution process be affected by the modification of any Process Recipes in storage, either by downloading or by local editing or by deletion, while that Process Recipe is being executed.
- The equipment must provide verification and validation of all downloaded Process Recipes.
- The equipment shall support at least one of the defined alternatives for Process Recipe management:
  - Stream 7 facilitates Process Recipe management for formatted and unformatted process programs.
  - Stream 15 facilitates Process Recipe management for E42 recipes.
  - Stream 13 is also utilized for Process Recipe management for large process programs and large E42 recipes.
  - Stream 19 and stream 13 facilitate Process Recipe management for E139 recipes.
- The equipment supplier shall document any restrictions on the length or text format of PPID. The maximum length allowed by equipment may be less than that allowed by SECS-II.
- Where E42 recipes are supported, the following requirements also apply:
- The variable PPFormat shall be provided to indicate to the user the messages supported by the equipment.
- The recipe and its attributes shall comply to the requirements for execution recipes as defined in SEMI E42, § 6.

7.7.5 *Scenarios for Process Programs*

7.7.5.1 Process Program Creation, Editing, or Deletion Process Program Created, Edited, or Deleted by Operator:

```
COMMENTS                          HOST    EQUIPMENT    COMMENTS

                                                       New process program created,
                                                       edited, or deleted by operator
                                                       at equipment.
                                                       PPChangeName = PPID
                                                       PPChangeStatus = 1 (Created)
                                                                      = 2 (Edited)
                                                                      = 3 (Deleted)
                                                       [IF] CEID for Process Program
                                                       Change Event enabled
                                                       [THEN]
                                        <--S6,F11      Send Event Report
Event Report Acknowledge          S6,F12-->
                                                        [END_IF]
```

LETTER BALLOT

7.7.5.2 Process Program Deletion by the Host:

| COMMENTS | HOST | EQUIPMENT | COMMENTS |
|---|---|---|---|
| Delete Process Program Send | S7,F17--> | | |
| | | <--S7,F18 | Delete Process Program Acknowledge.<br>The process program is removed from nonvolatile storage. |

7.7.5.3 Process Program Directory Request:

| COMMENTS | HOST | EQUIPMENT | COMMENTS |
|---|---|---|---|
| Current EPPD Request | S7,F19--> | | |
| | | <--S7,F20 | Current EPPD Data |

7.7.5.4 *Process Program Upload*

7.7.5.4.1 Host-Initiated Process Program Upload — Formatted:

| COMMENTS | HOST | EQUIPMENT | COMMENTS |
|---|---|---|---|
| Formatted Process Program Request | S7,F25--> | | |
| | | <--S7,F26 | Formatted Process Program Data |

7.7.5.4.2 Equipment-Initiated Process Program Upload — Formatted:

| COMMENTS | HOST | EQUIPMENT | COMMENTS |
|---|---|---|---|
| | | | [IF] Process program is multi-block<br>[THEN] |
| | | <--S7,F1 | Process Program Load Inquire |
| Process Program Load Grant | S7,F2--> | | |
| | | | [END_IF] |
| | | <--S7,F23 | Formatted Process Program Send |
| Formatted Process Program Acknowledge | S7,F24--> | | |

LETTER BALLOT

7.7.5.4.3 Host-Initiated Large Process Program Upload — Formatted:

| COMMENTS | HOST | EQUIPMENT | COMMENTS |
|---|---|---|---|
| Large Formatted Process Program Request | S7,F43--> | | |
| | | <--S7,F44 | Large Formatted Process Program Acknowledge<br><br>Acknowledge may be "accepted and will be performed later with completion signaled". When the equipment is ready to send: |
| | | <--S13,F1 | Send Data Set Send |
| Send Data Set Send Acknowledge | S13,F2--> | | |
| Open Data Set request | S13,F3--> | | |
| | | <--S13,F4 | Open Data Set Data |
| [START<br>Repeat until ACKC13 indicates error:<br>Read Data Set Request | S13,F5--> | | |
| [END] | | <--S13,F6 | Read Data set Data |
| | | | [IF]<br>CEID for upload event enabled<br>[THEN]<br>[IF]<br>ACKC13 is "ERROR: END OF DATA"<br>[Then] |
| Event Report Acknowledge | S6,F12--> | <--S6,F11 | Send Event Report "successful upload"<br>[ELSE] |
| Event Report Acknowledge | S6,F12--> | <--S6,F11 | Send Event Report "bad upload" |
| | | | [END_IF]<br>[END_IF] |
| Close Data Set Send | S13,F7--> | | |
| | | <--S13,F8 | Close Data Set Acknowledge |

LETTER BALLOT

7.7.5.4.4 Equipment-Initiated Large Process Program Upload — Formatted:

| COMMENTS | HOST | EQUIPMENT | COMMENTS |
|---|---|---|---|
| | | <--S7,F39 | Large Formatted Process Program Send |
| Large Formatted Process Program Acknowledge | S7,F40--> | | |
| | | <--S13,F1 | Send Data Set Send |
| Send Data Set Send Acknowledge | S13,F2--> | | |
| Open Data Set request | S13,F3--> | | |
| | | <--S13,F4 | Open Data Set Data |
| [START] repeat until ACKC13 is error: | | | |
| Read Data Set Request | S13,F5--> | | |
| [END] | | <--S13,F6 | Read Data set Data |
| | | | [IF] CEID for upload event enabled [THEN] [IF] ACKC13 is "ERROR: END OF DATA" [Then] |
| | | <--S6,F11 | Send Event Report "successful upload" [ELSE] |
| Event Report Acknowledge | S6,F12--> | | |
| | | <--S6,F11 | Send Event Report "bad upload" |
| Event Report Acknowledge | S6, F12--> | | |
| | | | [END_IF] [END_IF] |
| Close Data Set Send | S13,F7--> | | |
| | | <--S13,F8 | Close Data Set Acknowledge |

7.7.5.4.5 Host-Initiated Process Program Upload — Unformatted:

| COMMENTS | HOST | EQUIPMENT | COMMENTS |
|---|---|---|---|
| Process Program Request | S7,F5--> | | |
| | | <--S7,F6[1] | Process Program Data |

[1] If the process program does not exist, a zero-length list will be sent.

LETTER BALLOT

7.7.5.4.6 Equipment-Initiated Process Program Upload — Unformatted:

| COMMENTS | HOST | EQUIPMENT | COMMENTS |
|---|---|---|---|
| | | | [IF] Process program is multi-block<br>[THEN] |
| | | <--S7,F1 | Process Program Load Inquire |
| Process Program Load Grant | S7,F2--> | | |
| | | | [END_IF] |
| | | <--S7,F3 | Process Program Send |
| Process Program Acknowledge | S7,F4--> | | |

7.7.5.4.7 Host-Initiated Large Process Program Upload — Unformatted:

| COMMENTS | HOST | EQUIPMENT | COMMENTS |
|---|---|---|---|
| Large Process Program Request | S7,F41--> | | |
| | | <--S7,F42 | Large Process Program Acknowledge |
| | | | Acknowledge may be "accepted and will be performed later with completion signaled". When the equipment is ready to send: |
| | | <--S13,F1 | Send Data Set Send |
| Send Data Set Send Acknowledge | S13,F2--> | | |
| Open Data Set request | S13,F3--> | | |
| | | <--S13,F4 | Open Data Set Data |
| [START]<br>Repeat until ACKC13 indicates error: | | | |
| Read Data Set Request | S13,F5--> | | |
| [END] | | <--S13,F6 | Read Data set Data<br>[IF]<br>CEID for upload event enabled<br>[THEN]<br>[IF]<br>ACKC13 is "ERROR: END OF DATA"<br>[Then] |
| | | <-- 6,F11 | Send Event Report "successful upload" |
| Event Report Acknowledge | S6,F12--> | | |
| | | <--S6,F11 | [ELSE]<br>Send Event Report "bad upload" |
| Event Report Acknowledge | S6,F12--> | | |
| | | | [END_IF]<br>[END_IF] |
| Close Data Set Send | S13,F7--> | | |
| | | <--S13,F8 | Close Data Set Acknowledge |

7.7.5.4.8 Equipment-Initiated Large Process Program Upload — Unformatted:

| COMMENTS | HOST | EQUIPMENT | COMMENTS |
|---|---|---|---|
| | | <--S7,F37 | Large Process Program Send |
| Large Process Program Acknowledge | S7,F38--> | | |
| | | <--S13,F1 | Send Data set Send |
| Send Data Set Send Acknowledge | S13,F2--> | | |
| Open Data Set request | S13,F3--> | | |
| | | <--S13,F4 | Open Data Set Data |
| [START] Repeat until ACKC13 is error: | | | |
| Read Data Set Request | S13,F5--> | | |
| [END] | | <--S13,F6 | Read Data set Data |
| | | | [IF] CEID for upload event enabled [THEN] [IF] ACKC13 is "ERROR: END OF DATA" [Then] |
| | | <--S6,F11 | Send Event Report |
| Event Report Acknowledge | S6,F12--> | | "successful upload" |
| | | | [ELSE] |
| | | <--S6,F11 | Send Event Report "bad upload" |
| Event Report Acknowledge | S6,F12--> | | [END_IF] [END_IF] |
| Close Data Set Send | S13,F7--> | | |
| | | <--S13,F8 | Close Data Set Acknowledge |

7.7.5.5 *Process Program Download*

NOTE 16: Formatted process programs must be verified immediately following any download.

7.7.5.5.1 While the Process Program Load Inquire/Grant transaction (S7,F1/F2) is required only for the transfer of multi-block process programs, its use is recommended prior to all host-initiated downloads. It provides a means of verifying process program size.

LETTER BALLOT

7.7.5.5.2 Host-Initiated Process Program Download — Formatted:

| COMMENTS | HOST | EQUIPMENT | COMMENTS |
|---|---|---|---|
| [IF] Process program is multi-block [THEN] | | | |
| Process Program Load Inquire | S7,F1[1]--> | | |
| | | <--S7,F2 | Process Program Load Grant |
| [ENDIF] | | | |
| Formatted Process Program Send | S7,F23--> | | |
| | | <--S7,F24 | Formatted Process Program Acknowledge Verify process program [IF] S7,F27 is multi-block [THEN] |
| | | <--S7,F29 | Process Program Verification Inquire |
| | S7,F30-> | | Process Program Verification Grant [END IF] |
| | | <--S7,F27 | Process Program Verification Send |
| Process Program Verification Acknowledge | S7,F28--> | | |

7.7.5.5.3 Equipment-Initiated Process Program Download — Formatted:

| COMMENTS | HOST | EQUIPMENT | COMMENTS |
|---|---|---|---|
| | | <--S7,F25 | Formatted Process Program Request |
| Formatted Process Program Data | S7,F26--> | | |
| | | | [IF] S7,F27 is multi-block [THEN] |
| | | <--S7,F29 | Process Program Verification Inquire |
| | S7,F30--> | | Process Program Verification Grant [END IF] |
| | | <--S7,F27 | Process Program Verification Send |
| Process Program Verification Acknowledge | S7,F28--> | | |

[1] S7,F1 should be used only to request permission to transfer a multi-block formatted or unformatted process program. It should not be used to select a process program. For selecting a process program for execution, the remote command PP-SELECT should be used.

LETTER BALLOT

7.7.5.5.4 Host-Initiated Large Process Program Download — Formatted:

| COMMENTS | HOST | EQUIPMENT | COMMENTS |
|---|---|---|---|
| Large Formatted Process Program Send | S7,F39--> | | |
| | | <--S7,F40 | Large Formatted Process Program Acknowledge |
| | | <--S13,F3 | Open Data Set Request |
| Open Data Set Data | S13,F4--> | | |
| | | | [START] repeat until ACKC13 is error: |
| | | <--S13,F5 | Read Data Set Request |
| Read Data Set Data | S13,F6--> | | |
| | | | [END] |
| | | <--S13,F7 | Close Data Set Send |
| Close Data Set Acknowledge | S13,F8--> | | |
| | | | [IF] S7,F27 is multi-block [THEN] |
| | | <--S7,F29 | Process Program Verification Inquire |
| Process Program Verification Grant | S7,F30--> | | |
| | | | [END_IF] |
| | | <--S7,F27 | Process Program Verification Send |
| Process Program Verification Acknowledge | S7,F28--> | | |

LETTER BALLOT

7.7.5.5.5 Equipment-Initiated Large Process Program Download — Formatted:

| COMMENTS | HOST | EQUIPMENT | COMMENTS |
|---|---|---|---|
| | | <--S7,F43 | Large Formatted Process Program |
| Large Formatted Process Program Acknowledge | S7,F44--> | | Program Request |
| | | <--S13,F3 | Open Data Set Request |
| | S13,F4--> | | |
| Open Data Set Data | | | [START] repeat until ACK13 is error: |
| | | <--S13,F5 | Read Data Set Request |
| | S13,F6--> | | |
| Read Data Set Data | | | [END] |
| | | <--S13,F7 | Close Data Set Send |
| | S13,F8--> | | |
| Close Data Set Acknowledge | | | [IF] S7,F27 is multi-block [THEN] |
| | | <--S7,F29 | Process Program Verification Inquire |
| | S7,F30--> | | |
| Process Program Verification Grant | | | [END_IF] |
| | | <--S7,F27 | Process Program Verification Send |
| | S7,F28--> | | |
| Process Program Verification Acknowledge | | | |

7.7.5.5.6 Host-Initiated Process Program Download — Unformatted:

| COMMENTS | HOST | EQUIPMENT | COMMENTS |
|---|---|---|---|
| [IF] Process program is multi-block [THEN] | | | |
| Process Program Load Inquire | S7,F1[21]--> | | |
| | | <--S7,F2 | Process Program Load Grant |
| [END_IF] | | | |
| Process Program Send | S7,F3--> | | |
| | | <--S7,F4 | Process Program Acknowledge |

7.7.5.5.7 Equipment-Initiated Process Program Download — Unformatted:

| COMMENTS | HOST | EQUIPMENT | COMMENTS |
|---|---|---|---|
| | | <--S7,F5 | Process Program Request |
| Process Program Send | S7,F6--> | | |

LETTER BALLOT

7.7.5.5.8 Host-Initiated Large Process Program Download — Unformatted:

| COMMENTS | HOST | EQUIPMENT | COMMENTS |
|---|---|---|---|
| Large Process Program Send | S7,F37--> | | |
| | | <--S7,F38 | Large Process Program Acknowledge |
| | | <--S13,F3 | Open Data Set Request |
| | S13,F4--> | | |
| Open Data Set Data | | | [START] repeat until ACKC13 is error: |
| | | <--S13,F5 | Read Data Set Request |
| | S13,F6--> | | |
| Read Data set Data | | | [END] |
| | | <--S13,F7 | Close Data Set Send |
| | S13,F8--> | | |
| Close Data Set Acknowledge | | | |
| | | | [IF] S7,F27 is multi-block [THEN] Process Program Verification |
| | | <--S7,F29 | Inquire |
| Process Program Verification Grant | S7,F30--> | | [END_IF] |
| | | <--S7,F27 | Process Program Verification Send |
| Process Program Verification Acknowledge | S7,F28--> | | |

LETTER BALLOT

7.7.5.5.9 Equipment-Initiated Large Process Program Download — Unformatted:

| COMMENTS | HOST | EQUIPMENT | COMMENTS |
|---|---|---|---|
| | | <--S7,F41 | Large Process Program Request |
| Large Process Program Program Acknowledge | S7,F42--> | | |
| | | <--S13,F3 | Open Data Set Request |
| Open Data Set Data | S13,F4--> | | |
| | | | [START] repeat until ACKC13 is error: |
| | | <--S13,F5 | Read Data Set Request |
| Open Data Set Data | S13,F6--> | | |
| | | | [END] |
| | | <--S13,F7 | Close Data Set Send |
| Close Data Set Acknowledge | S13,F8--> | | |
| | | | [IF] S7,F27 is multi-block [THEN] |
| | | <--S7,F29 | Process Program Verification Inquire |
| Process Program Verification Grant | S7,F30--> | | [END_IF] |
| | | <--S7,F27 | Process Program Verification Send |
| Process Program Verification Acknowledge | S7,F28--> | | |

7.7.6 *Scenarios for Recipes*

7.7.6.1 *Recipe Creation, Editing, or Deletion*

7.7.6.1.1 Recipe Created by Operator:

| COMMENTS | HOST | EQUIPMENT | COMMENTS |
|---|---|---|---|
| | | | New recipe created by operator at equipment<br>RcpChangeName = RCPID<br>RcpChangeStatus = 1 (Created)<br>[IF] CEID for New Execution Recipe Event enabled [THEN] |
| | | <--S6,F11 | Send Event Report |
| Event Report Acknowledge | S6,F12--> | | |
| | | | [END_IF] |

LETTER BALLOT

7.7.6.1.2 Recipe Edited or Deleted by Operator:

| COMMENTS | HOST | EQUIPMENT | COMMENTS |
|---|---|---|---|
| | | | New recipe edited, or deleted at equipment<br>RcpChangeName = RCPID<br>RcpChangeStatus = 2 (Modified) or 5 (Deleted)<br>[IF] CEID for Execution Execution Recipe Change Event enabled<br>[THEN] |
| | | <--S6,F11 | Send Event Report |
| Event Report Acknowledge | S6,F12--> | | |
| | | | [END_IF] |

7.7.6.1.3 Recipe Deletion by the Host:

| COMMENTS | HOST | EQUIPMENT | COMMENTS |
|---|---|---|---|
| Recipe Delete Request | S15,F35--> | | |
| | | <--S15,F36 | Recipe Delete Acknowledge. The recipe is removed from non volatile storage. |

7.7.6.2 *Recipe Directory Request*

7.7.6.2.1 Host Requests a List of Identifiers of Currently Stored Recipes:

| COMMENTS | HOST | EQUIPMENT | COMMENTS |
|---|---|---|---|
| GetAttr Request | S14,F1--> | | |
| | | <--S14,F2 | GetAttr Data |

7.7.6.3 *Recipe Upload*

7.7.6.3.1 Host-Initiated Recipe Upload:

| COMMENTS | HOST | EQUIPMENT | COMMENTS |
|---|---|---|---|
| Recipe Upload Request | S15,F31--> | | |
| | | <--S15,F32 | Recipe Upload Data |

LETTER BALLOT

7.7.6.3.2 Host-Initiated Large Recipe Upload:

| COMMENTS | HOST | EQUIPMENT | COMMENTS |
|---|---|---|---|
| Large Recipe Upload Request | S15,F51--> | | |
| | | <--S15,F52 | Large Recipe Upload Acknowledge |
| | | | Acknowledge may be "accepted and will be performed later with completion signaled". When the equipment is ready to send: |
| | | <--S13,F1 | Send Data Set Send |
| Send Data Set Send Acknowledge | S13,F2--> | | |
| Open Data Set request | S13,F3--> | | |
| | | <--S13,F4 | Open Data Set Data |
| [START] repeat until ACKC13 is error: | | | |
| Read Data Set Request | S13,F5--> | | |
| [END] | | <--S13,F6 | Read Data Set Data |
| | | | [IF] CEID for upload event enabled [THEN] [IF] ACKC13 is "ERROR: END OF DATA" [Then] |
| | | <--S6,F11 | Send Event Report "successful upload" |
| Event Report Acknowledge | S6,F12--> | | [ELSE] |
| | | <--S6,F11 | Send Event Report "bad upload" |
| Event Report Acknowledge | S6,F12--> | | [END_IF] [END_IF] |
| Close Data Set Send | S13,F7--> | | |
| | | <--S13,F8 | Close Data Set Acknowledge |

LETTER BALLOT

7.7.6.3.3 Equipment-Initiated Recipe Upload:

| COMMENTS | HOST | EQUIPMENT | COMMENTS |
|---|---|---|---|
| | | | RCPCMD = Upload<br>[IF] multi-block request |
| | | <--S15,F1 | [THEN] Recipe Management Multi-block inquire |
| Recipe Management Multi-block Grant | S15,F2--> | | |
| | | | [END_IF] |
| | | <--S15,F21 | Recipe Action Request |
| Recipe Action Acknowledge | S15,F22--> | | |
| Host requests upload | S15,F31--> | | |
| | | <--S15,F32 | Recipe Upload Data |

7.7.6.3.4 Equipment-Initiated Large Recipe Upload:

| COMMENTS | HOST | EQUIPMENT | COMMENTS |
|---|---|---|---|
| | | | RCPCMD = Upload<br>[IF] multi-block request |
| | | <--S15,F1 | [THEN]<br>Recipe Management |
| Recipe Management Multi-block Grant | S15,F2--> | | Multi-block Inquire<br>[END_IF] |
| | | <--S15,F21 | Recipe Action Request |
| Recipe Action Acknowledge | S15,F22--> | | |
| | S15,F51--> | | |
| Large Recipe Upload Request | | | |
| | | <--S15,F52 | Large Recipe Upload Acknowledge |
| | | <--S13,F1 | Send Data Set Send |
| Send Data Set Send Acknowledge | S13,F2--> | | |
| Open Data Set Request | | <--S13,F3 | Open Data Set Data |
| | S13,F4--> | | |
| [START] repeat until ACKC13 is error: | | | |
| Read Data Set Request | S13,F5--> | | |
| [END] | | <--S13,F6 | Read Data Set Data |

LETTER BALLOT

```
                                                    [IF]
                                                    CEID for upload event enabled
                                                    [THEN]
                                                    [IF]
                                                    ACKC13 is "ERROR: END OF DATA"
                                                    [Then]
                                        <--S6,F11   Send Event Report "successful
                                                    upload"
Event Report Acknowledge            S6,F12-->       [ELSE]
                                        <--S6,F11   Send Event Report "bad upload"

Event Report Acknowledge            S6,F12-->       [END_IF]
                                                    [END_IF]

Close Data Set Send                 S13,F7-->
                                        <--S13,F8   Close Data Set Acknowledge
```

7.7.6.4 *Recipe Download*

7.7.6.4.1 The Recipe Management Multi-block Inquire/Grant transaction (S15,F1/F2) is required for the transfer of multi-block recipes. However, its use is recommended prior to all downloads, as it provides recipe size to the equipment.

NOTE 17: If the data item RCPOWCODE is TRUE in S15,F27, then a pre-existing recipe with the same identifier shall be overwritten.

7.7.6.4.2 Host-Initiated Recipe Download:

```
COMMENTS                            HOST    EQUIPMENT   COMMENTS

[IF]
Recipe is multi-block
[THEN]

Recipe Management Multi-
block Inquire                       S15,F1-->

                                        <--S15,F2   Recipe Management Multi-block
                                                    Grant
[END IF]
Recipe Download Request             S15,F27-->
                                                    [IF] RCPOWCODE = TRUE
                                                    delete any pre-existing recipe
                                                    with the same identifier before
                                                    storing new recipe
                                                    [END_IF]
                                        <--S15,F28  Recipe Download Acknowledge
```

LETTER BALLOT

7.7.6.4.3 Host-Initiated Large Recipe Download:

| COMMENTS | HOST | EQUIPMENT | COMMENTS |
|---|---|---|---|
| Large Recipe Download Request | S15,F49--> | | |
| | | | [IF] RCPOWCODE = TRUE<br>delete any pre-existing recipe with the same identifier before storing new recipe<br>[END_IF] |
| | | <--S15,F50 | Large Recipe Download Acknowledge |
| | | <--S13,F3 | Open Data Set Request |
| Open Data Set Data | S13,F4--> | | |
| | | | [START] repeat until ACKC13 is error: |
| | | <--S13,F5 | Read Data Set Request |
| Read Data Set Data | S13,F6--> | | |
| | | | [END] |
| | | <--S13,F7 | Close Data Set Send |
| Close Data Set Acknowledge | S13,F8--> | | |
| | | | [IF] multi-block request |
| | | <--S15,F1 | [THEN] |
| Recipe Management Multi-block Grant | S15,F2--> | | Recipe Management Multi-block Inquire<br>[END_IF] |
| | | <--S15,F53 | Recipe Verification Send |
| Recipe Verification Acknowledge | S15,F54--> | | |

LETTER BALLOT

7.7.6.4.4 Equipment-Initiated Recipe Download:

| COMMENTS | HOST | EQUIPMENT | COMMENTS |
|---|---|---|---|
| | | | RCPCMD = Download<br>[IF] multi-block request |
| | | <--S15,F1 | [THEN] Recipe Management Multi-block inquire |
| Recipe Management Multi-block Grant | S15,F2--> | | |
| | | | [END_IF] |
| | | <--S15,F21 | Recipe Action Request |
| Recipe Action Acknowledge<br>[IF]<br>Recipe is multi-block<br>[THEN] | S15,F22--> | | |
| Recipe Management Multi-block Inquire | S15,F1--> | | |
| | | <--S15,F2 | Recipe Management Multi-block Grant |
| [END IF] | | | |
| Recipe Download Request | S15,F27--> | | |
| | | | [IF] RCPOWCODE = TRUE<br>delete any pre-existing recipe with the same identifier before storing new recipe<br>[END_IF] |
| | | <--S15,F28 | Recipe Download Acknowledge |

LETTER BALLOT

7.7.6.4.5 Equipment-Initiated Large Recipe Download:

| COMMENTS | HOST | EQUIPMENT | COMMENTS |
|---|---|---|---|
| | | | RCPCMD = Download<br>[IF] multi-block request |
| | | <--S15,F1 | [THEN] |
| Recipe Management Multi-block Grant | S15,F2--> | | Recipe Management Multi-block Inquire<br>[END_IF] |
| | | <--S15,F21 | Recipe Action Request |
| Recipe Action Acknowledge | S15,F22--> | | |
| Large Recipe Download Requests | S15,F49--> | | [IF] RCPOWCODE = TRUE<br>delete any pre-existing recipe with the same identifier before storing new recipe<br>[END_IF] |
| | | <--S15,F50 | Large Recipe Download Acknowledge |
| Open Data Set Data | S13,F4--> | | |
| | | <--S13,F3 | Open Data Set Request |
| | | | START] repeat until ACKC13 is error: |
| | | <--S13,F5 | Read Data Set Request |
| Read Data Set Data | S13,F6--> | | |
| | | | [END] |
| | | <--S13,F7 | Close Data Set Send |
| Close Data Set Acknowledge | S13,F8--> | | |
| | | | [IF] multi-block request<br>[THEN] |
| | | <--S15,F1 | Recipe Management Multi-block Inquire |
| Recipe Management Multi-block Grant | S15,F2--> | | [END_IF] |
| | | <--S15,F53 | Recipe Verification Send |
| Recipe Verification Acknowledge | S15,F54--> | | |

7.7.6.5 *Recipe Verification*

7.7.6.5.1 Host Requests Equipment to Verify a Recipe that it Has Stored:

| COMMENTS | HOST | EQUIPMENT | COMMENTS |
|---|---|---|---|
| Recipe Verify Request | S15,F29--> | | |
| | | | Equipment verifies recipe |
| | | <--S15,F30 | Recipe Verify Data |

7.7.7 *E139* — Recipe and Parameter (RaP) Management Scenarios

7.7.7.1 See SEMI E139 for messaging scenarios related to E139 recipes.

LETTER BALLOT

7.8 *Material Movement* — The material movement capability includes the physical transfer of material (WIP, tools, expendable materials, etc.) between equipment, buffers, and storage facilities. The transfer of material can be performed by operators, AGV robots, tracks, or dedicated fixed material handling equipment.

7.8.1 *Purpose* — This capability is limited in implementation, serving to notify the host of the appearance or removal of material at the equipment's ports.

7.8.2 *Definitions*

7.8.2.1 *Port* — A point or area on the equipment at which a change in equipment ownership of material may occur.

7.8.3 *Description* — This capability consists of alerting the host whenever material is sent or received from any of the ports on the equipment. Event-specific information, such as port identification and material identification, also may be useful, but definition of these and other related DVVAL's are left to the implementation.

7.8.4 *Requirements* — The equipment must supply two CEIDs, one to report when material is sent from any port and the other to report when material is received at any port.

7.8.5 *Scenario*

| **COMMENTS** | **HOST** | **EQUIPMENT** | **COMMENTS** |
|---|---|---|---|
| | | | Material is sent or received at an equipment port. |
| | | <--S6,F11 | Send collection event to host. |
| Host acknowledges. | S6,F12--> | | |

7.9 *Equipment Terminal Services* — Equipment Terminal Services allows the host to display information on the equipment's display device or the operator of the equipment to send information to the host.

7.9.1 *Purpose* — Equipment Terminal Services allows the factory operators to exchange information with the host from their equipment workstations.

7.9.2 *Definitions*

7.9.2.1 *Message Recognition*— A positive action by the equipment operator indicating the operator has viewed the text of a host initiated message.

7.9.3 *Detailed Description* — The equipment must be capable of displaying information passed to it by the host for the operator's attention. The information, or an indication of a message, must remain on the equipment's display until the operator indicates message recognition. Message recognition results in a collection event that informs the host that the operator has actually viewed the information.

7.9.3.1 The equipment must be capable of passing information to the host that has been entered from the operator's equipment console. This information is intended for host applications and is not processed by the equipment.

7.9.3.2 The equipment has no responsibility for interpreting any of the data passed to or from the host using this method.

7.9.4 *Requirements*

- Any new Terminal Display message sent by the host shall overwrite an unrecognized message at the same equipment terminal.
- The equipment must provide a display device capable of displaying at least 160 characters to the operator.
- The equipment must provide a mechanism for displaying information sent to it by the host.
- The equipment must provide an indicator to notify the operator when an unrecognized message is present.
- The equipment must provide a mechanism for the operator to indicate message recognition (e.g., push button, terminal function).
- The equipment must provide a means for alpha numeric data entry that can be used by the operator.

LETTER BALLOT

- The equipment must support operator entry of at least 160 characters per message.
- The equipment must have a mechanism to send operator-entered messages to the host.
- The equipment must support single-block messages as a minimum. Support of multi-block messages is optional.
- A Terminal Display message received by the equipment with a zero length TEXT data item shall be accepted and replace any previous unrecognized message, but shall not itself be considered an unrecognized message. This provides a method of clearing an unrecognized message and turning off the unrecognized message indicator.

7.9.5 *Scenarios*

7.9.5.1 Host Sends Information to an Equipment's Display Device:

| COMMENTS | HOST | EQUIPMENT | COMMENTS |
|---|---|---|---|
| Host sends textual information to equipment for display to the operator on terminal x. | S10,F3--> | | |
| | | <--S10,F4 | Equipment acknowledges request to display text. (Equipment sets unrecognized message indicator.) Operator indicates message recognition. (Equipment clears unrecognized message indicator.) |
| | | <--S6,F11 | Message recognition event. (See § 5.3.1, Event Data Collection, for details.) |
| Host acknowledges Optional: | S6,F12--> | | |
| | | <--S10,F1 | Operator responds with text via terminal x. |
| Host acknowledges receipt of operator text. | S10,F2--> | | |

LETTER BALLOT

7.9.5.2 Host Sends Information to an Equipment's Display Device and Then Overwrites the Information Before Operator Recognizes Message:

| COMMENTS | HOST | EQUIPMENT | COMMENTS |
|---|---|---|---|
| Host sends textual information to equipment for display to the operator on terminal x. | S10,F3--> | | |
| | | <--S10,F4 | Equipment acknowledges request to display text. (Equipment sets unrecognized message indicator.) |
| Host sends textual information to equipment for display to the operator on terminal x. This message overwrites the first one sent by the host since it is still unrecognized. | S10,F3--> | | |
| | | <--S10,F4 | Equipment acknowledges request to display text. (Equipment sets unrecognized message indicator.) |
| | | | Operator indicates message recognition. (Equipment clears unrecognized message indicator.) |
| | | <--S6,F11 | Message recognition event. (See § 5.3.1, Event Data Collection, for details.) |
| Host acknowledges | S6,F12--> | | |

7.9.5.3 Operator Sends Information to the Host:

| COMMENTS | HOST | EQUIPMENT | COMMENTS |
|---|---|---|---|
| | | <--S10,F1 | Operator sends textual information via equipment terminal x. |
| Host acknowledges receipt of operator initiated message. | S10,F2--> | | |
| Optional: | | | |
| Host responds with information for display to the operator on terminal x. | S10,F3--> | | |
| | | <--S10,F4 | Equipment acknowledges receipt of request to display text. (Equipment sets unrecognized message indicator.) |
| | | | Operator indicates message recognition. |
| | | <--S6,F11 | Message recognition event. (See Event data collection for details.) |
| Host acknowledges | S6,F12--> | | |

LETTER BALLOT

7.9.5.4 Host Sends a Multi-Block Display Message:

| COMMENTS | HOST | EQUIPMENT | COMMENTS |
|---|---|---|---|
| Send information | S10,F5--> | | |
| | | <--S10,F6 | Accepted or denied.<br>[IF]<br>Message from host is multi-block and multi-block is not supported by the equipment,<br>[THEN] |
| | | <--S10,F7 | Send Multi-block Not Allowed.<br>[END_IF] |

7.10 *Error Messages*

7.10.1 *Purpose* — Error messages provide the host with information describing the reason for a particular message or communication fault detected by the equipment.

7.10.2 *Definitions*

7.10.2.1 *Communication Fault* — Refer to § 3 for the definition.

7.10.2.2 *Message Fault* — Refer to § 3 for the definition.

7.10.3 *Detailed Description* — The equipment must inform the host that it cannot process a message due to an incorrect:

- device ID,
- message stream type,
- message function,
- message format, or
- data format.

7.10.3.1 The equipment must inform the host if the message has more data than it can handle.

7.10.3.2 The equipment must inform the host if the equipment's transaction timer expires.

7.10.3.3 The equipment shall treat the above conditions as application-level errors and shall not take any further action on any message in error.

7.10.3.4 Error messages are invoked whenever the equipment detects communication or message faults.

7.10.4 *Requirements* — Support of all Stream 9 messages.

LETTER BALLOT

7.10.5 *Scenarios*

7.10.5.1 Message Fault Due to Unrecognized Device ID:

| COMMENTS | HOST | EQUIPMENT | COMMENTS |
|---|---|---|---|
| Host sends a message. | Sx,Fy--> | | |
| | | | Equipment detects an unrecognized device ID within the message from the host. |
| | | <--S9,F1 | Equipment reports to the host that an "unrecognized device ID" was detected in the received message. |

7.10.5.2 Message Fault Due to Unrecognized Stream Type:

| COMMENTS | HOST | EQUIPMENT | COMMENTS |
|---|---|---|---|
| Host sends a message. | Sx,Fy--> | | |
| | | | Equipment detects an unrecognized stream type within the message from the host. |
| | | <--S9,F3 | Equipment reports to the host that an "unrecognized stream type" was detected in the received message. |

7.10.5.3 Message Fault Due to Unrecognized Function Type:

| COMMENTS | HOST | EQUIPMENT | COMMENTS |
|---|---|---|---|
| Host sends a message. | Sx,Fy--> | | |
| | | | Equipment detects an unrecog-nized function type within the message from the host. |
| | | <--S9,F5 | Equipment reports to the host that an "unrecognized function type" was detected in the received message. |

7.10.5.4 Message Fault Due to Illegal Data Format:

| COMMENTS | HOST | EQUIPMENT | COMMENTS |
|---|---|---|---|
| Host sends a message. | Sx,Fy--> | | |
| | | | Equipment detects illegal data format within the message from the host. |
| | | <--S9,F7 | Equipment reports to the host that "illegal data format" was detected in the received message. |

7.10.5.5 Communication Fault Due to Transaction Timer Timeout:

| COMMENTS | HOST | EQUIPMENT | COMMENTS |
|---|---|---|---|
| | | | Equipment does not receive an expected reply message from the host and a transaction timer timeout occurs. |
| | | <--S9,F9 | Equipment reports to the host that a transaction timer timeout occurred. |

7.10.5.6 Message Fault Due to Data Too Long:

| COMMENTS | HOST | EQUIPMENT | COMMENTS |
|---|---|---|---|
| Host sends a message. | Sx,Fy--> | | |
| | | | Equipment detects that the message from the host contains more data than it can handle. |
| | | <--S9,F11 | Equipment reports to the host that "data too long " was detected in the received message. |

7.10.5.7 Communication Fault Due to Conversation Timeout:

| COMMENTS | HOST | EQUIPMENT | COMMENTS |
|---|---|---|---|
| Host sends a message. | Sx,Fy--> | | |
| | | <--Sx,Fy+1 | Equipment sends reply. Equipment is now expecting a specific message from the host as a result of the previous transaction. Equipment has not received the expected message from the host and a conversation timeout occurs. |
| | | <--S9,F13 | Equipment reports to the host that a conversation timeout occurred. |

7.11 *Clock* — The clock capability enables host management of time-related activities and occurrences associated with the equipment and across multiple pieces of equipment.

7.11.1 *Purpose* — The purpose of the clock capability is to enable time stamping of collection event and alarm reports. Time stamping is useful for resolving relative order of event/alarm occurrences and scheduling of equipment activities by the host.

7.11.1.1 The ability for the host to instruct the equipment to set an internal clock to a specified time value, and for the equipment to request the current date and time, is needed for effective time management and synchronization between host and equipment.

LETTER BALLOT

7.11.2 *Definitions*

7.11.2.1 *Clock* — Clock is a status variable containing the current value of time at the equipment. Clock may be included in report definitions and/or queried separately by the host. See SEMI E5 for a full definition of this variable data item and its required formatting.

7.11.2.2 *TIME* — TIME is a data item contained in messages used by the host to set time at the equipment and by the equipment or host to request the current time from the other. See SEMI E5 for a full definition of this data item.

7.11.3 *Detailed Description* — The clock capability assumes the existence of a relative time reference on the equipment. This time reference is used as a basis for updating the time value of an equipment status variable called 'Clock'. The time reference must reflect the current time to within a resolution range of seconds to centiseconds (refer to the format for Clock in the SEMI E5 Standard). The purpose of time stamping with centiseconds is to resolve the order in which nearly simultaneous events occur rather than to provide a more precise record of the time of day at which they occurred. Where more than one event occurs within a given period of clock resolution, the centiseconds reported in the time stamps for these events must reflect the actual order in which the events were detected. Equipment with a clock resolution of less than a second should report centiseconds. Otherwise, centiseconds should be assigned to reflect the relative order in which events were detected. Equipment unable to resolve time to less than a second and unable to reflect the relative order in which events were detected may report centiseconds as '00'.

7.11.3.1 The host employs the 'Date and Time Set Request' message (S2,F31) to initialize the value of Clock to the value contained in the TIME data item. Similarly, the equipment may employ the 'Date and Time Request' message (S2,F17) to obtain a new initialization time for Clock. As before, the value of TIME returned by the host is used to set Clock. Note that, in the event that the precision of TIME is seconds and that for Clock is centiseconds, in both cases the initial value of Clock shall contain '00' for its Centisecond digits upon initialization. Additionally, for any field in TIME that is not supported by the equipment, the local value of this field is equipment dependent. For example, Equipment that cannot resolve time to less than a second might round or ignore centiseconds and always set the Centisecond field to '00'.

7.11.4 *Requirements*

- The resolution and update rate of the internal time reference must be sufficient to distinguish between two nearly simultaneous collection events and/or alarms.
- The equipment supplier shall provide documentation describing the resolution of the internal time reference.
- The equipment supplier shall provide documentation describing how centisecond values are assigned, including the case of unresolvable simultaneous events.

7.11.5 *Scenarios*

7.11.5.1 Equipment Requests TIME (Optional Scenario):

| COMMENTS | HOST | EQUIPMENT | COMMENTS |
|---|---|---|---|
| | | <--S2,F17 | Equipment requests a time value from the host. |
| Host responds with a TIME value S2,F18--> | | | |
| | | | Equipment sets its internal time reference to the value of TIME received from the host. |

LETTER BALLOT

7.11.5.2 Host Instructs Equipment to Set Time:

| COMMENTS | HOST | EQUIPMENT | COMMENTS |
|---|---|---|---|
| Host instructs equipment to set its time. | S2,F31--> | | |
| | | <--S2,F32 | Equipment sets its internal time reference to the value of TIME received from the host and acknowledges completion. |

7.11.5.3 Host Requests Equipment's Current Time Value:

| COMMENTS | HOST | EQUIPMENT | COMMENTS |
|---|---|---|---|
| Host requests equipment time. | S2,F17--> | | |
| | | <--S2,F18 | Equipment returns its internal time reference value to the host. |

7.12 *Spooling* — Spooling is a capability whereby the equipment can queue messages intended for the host during times of communication failure and subsequently deliver these messages when communication is restored. Spooling is limited to primary messages of user-selected streams.

7.12.1 *Purpose* — The purpose of spooling is to provide a method for retaining equipment message data that might otherwise be lost due to communication failure. The motive for producing this functionality is to retain valuable data used to track material and to improve product quality. The spooling capability fills a gap in the SEMI E5 Standard. In the past, without a spooling capability, the equipment has typically discarded messages that could not be delivered, or turned messaging off altogether. It is intended that the host initiate the spool unload process immediately following the reestablishment of communications.

7.12.2 *Definitions*

7.12.2.1 *MaxSpoolTransmit* — An equipment constant containing the maximum number of messages that the equipment shall transmit from the spool in response to an S6,F23 'Transmit Spooled Messages' request. If MaxSpoolTransmit is set to zero, no limit is placed on the messages sent from the spool. Multi-block inquire/grant messages are not counted in this total.

7.12.2.2 *OverWriteSpool* — A boolean equipment constant used to indicate to the equipment whether to overwrite data in the spool area or to discard further messages whenever the spool area limits are exceeded.

7.12.2.3 *Send Queue* — Refers to the queue into which equipment generated SECS messages are placed in preparation for transmission to the host.

7.12.2.4 *Spool* — The spool is an area of nonvolatile storage in which the equipment stores certain messages that cannot be delivered to the host (when the equipment is in the NOT COMMUNICATING substate of COMMUNICATIONS ENABLED). The spool area can be thought of as a sequential 'ring' buffer. The term spool is also used to denote the action of placing messages into the spool area.

7.12.2.5 *SpoolCountActual* — A status variable used to keep a count of the messages actually stored in the equipment's spool area. Multi-block inquire/grant messages are not spooled and not included in this count.

7.12.2.6 *SpoolCountTotal* — A status variable used to keep a count of the total number of primary messages directed to the spool, regardless of whether placed or currently retained in the spool. Multi-block inquire/grant messages are not spooled and not included in this count.

7.12.2.7 *SpoolFullTime* — A status variable containing the timestamp when the spool last became full. If the spool was not filled during the last spooling period, this will contain a time value prior to the current SpoolStartTime.

LETTER BALLOT

7.12.2.8 *SpoolStartTime* — A status variable containing the timestamp from when spooling was last activated.

7.12.3 *Description*

7.12.3.1 *Spooling State Model* — There are two major states of spooling: SPOOL INACTIVE and SPOOL ACTIVE. SPOOL ACTIVE has two components: SPOOL UNLOAD and SPOOL LOAD. These are each broken into substates. A description of all spooling states, substates, and applicable state transitions follows. The POWER OFF and POWER ON parent states are common to all equipment subsystems. They are shown here to illustrate the retention of spooling context during a power down situation.

NOTE 18: Disabling SECS communications does not affect the current spooling state since no messages are generated until communications are subsequently enabled. Spooling is effectively frozen in this case.

7.12.3.2 *POWER OFF*

7.12.3.2.1 The equipment has lost power for any reason (e.g., power failure, power switch set to off).

7.12.3.3 *POWER ON*

7.12.3.3.1 The equipment is powered up.

7.12.3.4 *SPOOL INACTIVE*

7.12.3.4.1 This is the normal operating mode. No spooling occurs. The spool area is empty. Primary SECS-II messages are transmitted normally.

7.12.3.5 *SPOOL ACTIVE*

7.12.3.5.1 All primary SECS-II messages ready for sending and for which spooling is enabled (see S2,F43) are directed to the spool area. All other primary messages, except Stream 1, are discarded. The equipment shall attempt to send any secondary messages that are generated and discard these messages should the attempt to send fail.

7.12.3.6 Spool state transitions from SPOOL INACTIVE to SPOOL ACTIVE if the communications state changes from COMMUNICATING to NOT COMMUNICATING (Communication State Transition Table, #14) or from WAIT CRA to WAIT DELAY (Communication STATE Transition Table, #6).

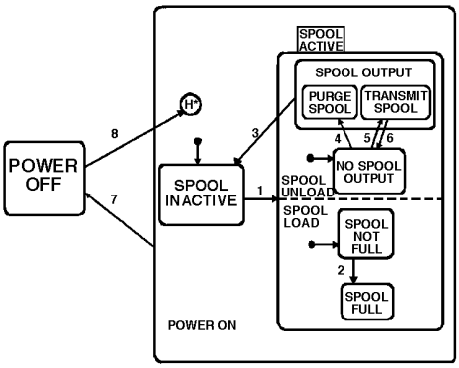



**Figure 12**
**Spooling State Diagram**

7.12.3.7 Once communications are established, the host must initiate the spool unload sequence to restore full functionality (see below). Since the equipment will deliver secondary messages, the host may inquire for information or send commands as needed.

7.12.3.8 The SPOOL ACTIVE state has two AND states: SPOOL LOAD and SPOOL UNLOAD. This means that they operate independently, though sharing data and some state change stimuli. See § 4.3 for explanation of state model terms and notation.

7.12.3.9 *SPOOL LOAD*

7.12.3.9.1 The SPOOL LOAD component enters messages into the spool area. It is divided into two substates: SPOOL NOT FULL and SPOOL FULL. SPOOL NOT FULL is the default entry substate of the parent state SPOOL LOAD.

7.12.3.10 *SPOOL NOT FULL*

7.12.3.10.1 As primary SECS-II messages are directed to the spool area, the equipment shall 'write' the SECS-II message to the end of the spool. Status variables SpoolCountTotal and SpoolCountActual shall be incremented each time a message is placed in the spool area.

7.12.3.11 *SPOOL FULL*

7.12.3.11.1 In this state, all of the allocated spooling area is filled. Choice of the following options shall be controlled by the setting of a Boolean equipment constant called 'OverWriteSpool'. The first message to be dealt with is that which could not be fit into the spool prior to transition from SPOOL NOT FULL (see transition table below).

7.12.3.12 *OverWriteSpool is True* — The equipment deletes as many of the 'oldest' records (e.g., SECS-II messages) contained in the spool area necessary to make space for the new message and then adds the message. Status variable SpoolCountTotal shall be incremented whenever a message is submitted to the spool area. Status variable SpoolCountActual shall be manipulated to keep an accurate count of the number of messages contained in the spool area. For example, if it is necessary to delete three messages in the spool area to spool one message, SpoolCountActual would have three subtracted and then one added to the total.

7.12.3.13 *OverWriteSpool is False* — Any subsequent primary messages shall be discarded. When such a message is discarded, SpoolCountTotal is incremented.

7.12.3.14 *SPOOL UNLOAD*

7.12.3.14.1 The SPOOL UNLOAD component of SPOOLACTIVE deals with movement of messages out of the spool. It has an active substate (SPOOL OUTPUT) and a passive substate (NO SPOOL OUTPUT). NO SPOOL OUTPUT is the default entry substate, since the equipment is NOT COMMUNICATING at the time spooling is initiated. When communications between equipment and host are restored, there is an opportunity for the host to recover spooled messages. No action is taken until the host initiates the spool output process via the S6,F23 (Request Spooled Data). The host has the option to either receive the spooled messages (see substate TRANSMIT SPOOL) or discard all messages in the spool (see substate PURGE SPOOL).

7.12.3.15 *NO SPOOL OUTPUT*

7.12.3.15.1 In this state, no messages are removed from the spool.

7.12.3.16 *SPOOL OUTPUT*

7.12.3.16.1 The SPOOL OUTPUT state encompasses the removal of messages from the spool. Its substates are TRANSMIT SPOOL and PURGE SPOOL.

7.12.3.17 *TRANSMIT SPOOL*

7.12.3.17.1 The host elects to receive all messages contained in the spool area. The equipment is expected to keep track of the oldest record (i.e., message) within the spool area. When communications are re-established with the host and transmission of the spool area is started, the oldest record must be the first record transmitted, then the next oldest record, etc. There is no prioritization of messages to be sent from the spool.

7.12.3.17.2 As each spooled message is successfully transmitted to the host, it is removed from the spool area upon successful completion of the transaction. SpoolCountActual is decremented as each message is removed from the spool. The equipment shall transmit messages only from the spool area until all spooled messages have been completely transmitted to the host.

7.12.3.17.3 Flow control of the spool transmit process is achieved in two ways. First, only one open transaction on the equipment is allowed during spool unload. Thus, if a message requires a reply, the equipment shall wait for that reply before transmitting the next spooled message. Messages which require no reply may be transmitted sequentially as rapidly as the message transfer mechanism will allow.

7.12.3.17.4 The second flow control method is to allow the host to limit the maximum number of messages sent from the spool in response to the S6,F23 request. An equipment constant named MaxSpoolTransmit may be set by the host to achieve this behavior. If MaxSpoolTransmit is set to five, for example, the equipment will send the first five messages from the spool and then transition to the NO SPOOL OUTPUT state, awaiting the next S6,F23 request. There is no event report generated when MaxSpoolTransmit is reached. The host is responsible for determining this situation by (a) counting the messages received, (b) timing out waiting for the next message, (c) inquiring to the equipment for the curent value of the SpoolCountActual status variable, or (d) some combination of the above. If MaxSpoolTransmit is set to zero, the spool shall be transmitted completely in response to S6,F23.

7.12.3.17.5 Normal spooling continues during the spool transmit process. If the SPOOL LOAD component should transition to SPOOL FULL, it shall not have any effect on the SPOOL UNLOAD component. Once full, the spool cannot make the transition back to SPOOL NOT FULL except via the SPOOL INACTIVE state. Space made available due to the spool unload process shall not be used in this case.

7.12.3.17.6 When a multi-block message is to be transmitted from the spool, any required inquire/grant transaction shall be initiated. If the host's response denies permission to send the multi-block message, the equipment shall discard that message and continue with the transmit process. This sequence shall count as one message in the MaxSpoolTransmit count.

7.12.3.17.7 There is one area where SPOOL LOAD and SPOOL UNLOAD may interact: When the spool is full and OverWriteSpool is True. During the spool transmit process, spooled messages are being removed and new primary messages are being written to the spool. These new messages are overwriting the oldest messages available, unless the unload process has freed sufficient spool space. There is a possibility that the unload and overwrite processes may compete for control of the same message area. For example, if the spool holds messages ABCDE, with A oldest and E newest, A might be sent to the host, B (and the space from A) overwritten by the new message F, C sent to the host, D and E (and the space from message C) overwritten by G, etc. The loss of continuity may be 'disorienting' to the host program receiving the messages. It is expected that the unload process will be fast relative to the generation of new messages, so that this occurrence will be rare.

7.12.3.17.8 Should a communication failure occur during the spool transmit process, spooling shall continue as before the transmit process began. However, the spool unload sequence shall terminate (i.e., transition to NO SPOOL OUTPUT will occur—see transition table below).

7.12.3.18 *PURGE SPOOL*

7.12.3.18.1 The equipment shall discard all messages in the spool and, when the spool is empty, zero SpoolCountActual.

7.12.3.19 *Spooling State Transitions*

7.12.3.19.1 A table follows detailing all spooling state transitions as presented in the state transition diagram.

**Table 7 Spooling State Transition**

| # | *Current State* | *Trigger* | *New State* | *Action* | *Comment* |
|---|---|---|---|---|---|
| 1 | SPOOL INACTIVE | Communication state changes from COMMUNICAT-ING to NOT COMMUNICAT-ING or from WAIT CRA to WAIT DELAY and Enable Spool is true. | SPOOL ACTIVE | SpoolCountActual and SpoolCountTotal are initialized to zero. Any open transactions with the host are aborted. SpoolStart-Time (SV) is set to current time. Alert the operator that spooling is active. | The default state in each AND substate is entered. The message which could not be sent remains in the send queue and is dealt with in Spool Active state. The collection event Spooling Activated has occurred. |
| 2 | SPOOL NOT FULL | Message generate which will not fit into spool area. | SPOOL FULL | SpoolFullTime (SV) is set to current time. Alert the operator that spool is full. | The message which would not fit into the spooling area is dealt with after the transition. No collection event is generated. |

| # | *Current State* | *Trigger* | *New State* | *Action* | *Comment* |
|---|---|---|---|---|---|
| 3 | SPOOL OUTPUT | Spool area emptied. | SPOOL INACTIVE | Spooling process disabled. Alert the operator that spooling has been terminated. | The collection event Spooling Deactivated has occurred. Transition from the AND substate Spool Unload component occurs. |
| 4 | NO SPOOL OUTPUT | S6,F23 received w/RSDC = 1. | PURGE SPOOL | No action. | Initiates purging process. No collection event is generated since this is based on host request. |
| 5 | NO SPOOL OUTPUT | S6,F23 received w/RSDC = 0. | TRANSMIT SPOOL | No action. | Initiates message transmission from spool. No collection event is generated since this is based on host request. |
| 6 | TRANSMIT SPOOL | Communication failure or MaxSpoolTransmit reached. | NO SPOOL OUTPUT | Spool transmission process suspended. | If communications failure, the event Spool Transmit Failure has occurred. No collection event generated for MaxSpoolTransmit reached. |
| 7 | POWER ON | Equipment power source discontinued. | POWER OFF | No action. | Spooling context has been maintained in nonvolatile storage prior to this transition. |
| 8 | POWER OFF | Equipment power source restored. | POWER ON | Spooling context restored from nonvolatile memory. | If spooling were active prior to power down, it shall continue. If TRANSMIT SPOOL were active at powerdown, transition #6 is expected to follow since communications state is initially NOT COMMUNICATING. |

7.12.3.20 *Enabling Spooling* — The equipment shall provide the host with the ability to enable and disable Spooling for any message (except Stream 1 messages [i.e., S1,F1, S1,F13]) via the S2,F43/F44 transaction. Spooling may be enabled for an entire Stream, for individual messages within a stream, or for any combination of the two. Streams and Functions not referenced in this message are not spooled. Spooling can be totally disabled by sending an S2,F43 with a zero length list for the first item (see S2,F43 definition). In addition, the equipment shall provide an equipment constant, EnableSpooling, to allow setting the enable or disable of spooling.

NOTE 19: When EnableSpooling is false, the SPOOL state cannot transition from SPOOLINACTIVE to SPOOLACTIVE. Changing EnableSpooling does not change the spool state, purge the spool or change streams and functions enabled for spooling. Once the equipment is 'SPOOL ACTIVE', the host must initiate the spool unload sequence to restore full functionality even though 'Enable Spooling' has changed to false.

7.12.4 *Requirements* — The following items are required to support the spooling capability:

- At a minimum, the equipment shall reserve for spooling nonvolatile storage with sufficient capacity to store all of the primary SECS-II messages that would occur during a normal processing cycle.
- While spooling is enabled, the equipment shall reply to primary messages sent by the host with the appropriate secondary message.
- Secondary messages which cannot be delivered shall be discarded, never spooled.
- All spooling-related status variables and setup information (as per S2,F43) must be stored in nonvolatile memory along with any other information required for the potential unloading of the spool area after a power loss.
- Upon power-up, the equipment shall retain all spooling context from the time the equipment was last shutdown or reset. This means that spooling, if previously active, continues upon system power up.
- The Equipment must reject any message that attempts to set 'Spooling' for Stream 1.
- If a multi-block primary message need for the inquire/grant is to be sent during SPOOL ACTIVE, the message should be placed in the spool and the grant resolved during spool transmit.

LETTER BALLOT

7.12.5 *Scenarios*

7.12.5.1 Define the Set of Messages to be Spooled:

7.12.5.1.1 This Scenario is used to set up the list of messages that the equipment should spool (or by defining none, to disable spooling).

| COMMENTS | HOST | EQUIPMENT | COMMENTS |
|---|---|---|---|
| Host defines messages to be spooled in case of communications failure. | S2,F43--> | | |
| | | <--S2,F44 | Equipment acknowledges setup. |

7.12.5.2 Define the Maximum Number of Messages to Send in Response to S6,F23:

7.12.5.2.1 This Scenario sets the value of the equipment constant MaxSpoolTransmit.

| COMMENTS | HOST | EQUIPMENT | COMMENTS |
|---|---|---|---|
| Host sends value for equipment constant MaxSpoolTransmit. | S2,F15--> | | |
| | | <--S2,F16 | Equipment acknowledges equipment constant change. |

7.12.5.3 Request or Delete Spooled Data (MaxSpoolTransmit = 0):

7.12.5.3.1 This Scenario is used to initiate the transfer of the spooled data from the equipment to the host or to purge the spool.

| COMMENTS | HOST | EQUIPMENT | COMMENTS |
|---|---|---|---|
| Communications were lost and then re-established. | | | |
| Host requests data that includes spool-related status variables. | S1,F3--> | | |
| | | <--S1,F4 | Send status data. |
| NOTE: S1,F3 is one of various methods that could be used. Request or delete spooled data. | S6,F23--> | | |
| | | <--S6,F24 | Request spooled data acknowledgement.<br>[IF] RSDC = 0 (Spool data requested.)<br>[THEN]<br>The appropriate Streams and Functions are used to transmit the spooled data to the host.<br>[ELSE_IF] RSDC = 1<br>[THEN]<br>Spool data discarded.<br>[END_IF] |
| | | <--S6,F11 | Spooling Deactivated event report sent. |
| Acknowledge | S6,F12--> | | |

LETTER BALLOT

7.12.5.4 Request or Delete Spooled Data (MaxSpoolTransmit > 0):

7.12.5.4.1 This Scenario shows the affect of MaxSpoolTransmit < SpoolCountActual on the Spool Transmit process. For the purpose of illustration, the value of MaxSpoolTransmit is 5 and the SpoolCountActual is 8 (at the time communications are re-established). No messages are added to the Spool during the transmit process.

| COMMENTS | HOST | EQUIPMENT | COMMENTS |
|---|---|---|---|
| | Communications were lost and then re-established. | | |
| Host requests data that includes spool-related status variables. | S1,F3--> | | |
| | | <--S1,F4 | Send status data (e.g., SpoolCountActual = 8, MaxSpool-Transmit = 5). |
| Host requests spooled data (RSDC=0). | S6,F23--> | | |
| | | <--S6,F24 | Request spooled data acknowledgement. The five oldest messages in the Spool are transmitted to the host. Spooling remains active. |
| Host recognizes that MaxSpoolTransmit is reached. | | | |
| Host requests additional spooled data (RSDC = 0). | S6,F23--> | | |
| | | <--S6,F24 | Request spooled data acknowledgement. The three remaining messages are transmitted from the spool. |
| | | <--S6,F11 | Spooling Deactivated event report sent. |
| Acknowledge | S6,F12--> | | |

7.13 *Control* — The control-related capabilities allow for configuration and manipulation of the control state model. In this way the host and/or user may modify the equipment's control-related behavior.

7.13.1 *Purpose* — This section complements the CONTROL state model description found in § 4.5. It defines the requirements for implementation of this model.

7.13.2 *Definitions* — None.

7.13.3 *Description*

7.13.3.1 *Control Configuration* — The control state model has two areas of configuration. The first area is related to default entry states of the state model. Upon system initialization, the system must activate either the ON-LINE or OFF-LINE state. Upon entry to OFF-LINE, the system must in turn activate one of the substates of OFF-LINE (EQUIPMENT OFF-LINE, ATTEMPT ON-LINE, or HOST OFF-LINE). In both these cases, the user shall configure the equipment to make the choices appropriate to that factory. Entry to the ON-LINE state also involves a choice of substates. In this case, the equipment reads the front panel REMOTE/LOCAL switch to determine the appropriate state.

7.13.3.1.1 The second area of configuration involves the transition to be made if the ON-LINE attempt should fail. The model may be set to transition to either HOST OFF-LINE or to EQUIPMENT OFF-LINE should the S1,F1 transaction be terminated unsuccessfully. Choosing HOST OFF-LINE allows the host to cause the equipment to transition to ON-LINE when the host becomes ready. This is accomplished via the message S1,F17 (see below).

7.13.3.2 *Changing Control State* — In the control state model, both the operator and the host can affect the control state. The operator retains ultimate authority to set the equipment OFF-LINE by means of an OFF-LINE switch

mechanism. The operator also can cause the equipment to attempt to go ON-LINE. Under some circumstances, the host can initiate the transition to ON-LINE.

7.13.3.2.1 If the operator requests ON-LINE, the equipment will send an S1,F1 to the host. The host may confirm ON-LINE with an S1,F2 or deny ON-LINE by sending an S1,F0.[1]

7.13.3.2.2 When the equipment is ON-LINE, the host may request that it transition to OFF-LINE. It will transition into the HOST OFF-LINE substate. When the equipment HOST OFF-LINE state is active, the host may request that it transition to ON-LINE. The combination of these two allow the host to cycle the equipment between ON-LINE and OFF-LINE.

7.13.3.2.3 Only the operator may change the ON-LINE substate (REMOTE or LOCAL).

7.13.4 *Requirements*

- The equipment shall supply a method for configuring the default CONTROL state to be activated upon system initialization. The choice of states must be among ATTEMPT ON-LINE, EQUIPMENT OFF-LINE, HOST OFF-LINE, and ON-LINE.
- The equipment shall supply a method for configuring which state should be activated when the attempt to go ON-LINE fails. The option is a transition to either the HOST OFF-LINE state or the EQUIPMENT OFF-LINE state.
- The equipment shall supply a momentary switch which will initiate the transition to OFF-LINE and another which will begin the process to go ON-LINE. Discrete position switches shall not be used. These should be designed so that they may not be actuated simultaneously. The switch may be mounted on the front panel or be available via keyboard input at the operator console.
- The equipment shall supply a discrete two-position switch which the operator may use to indicate the desired substate for ON-LINE (i.e., REMOTE or LOCAL). The switch may be mounted on the front panel or be available via keyboard input at the operator console. If implemented in software, this setting shall be retained in non-volatile storage.
- The equipment shall supply an indicator on the front panel which displays the full identification of the current CONTROL state/substate (e.g., OFF-LINE/ATTEMPT ON-LINE). This may be accomplished either with labelled display lights or via the operator console display. It is recommended that this indicator be visible at all times.
- The equipment shall supply a status variable that contains the current state/substate of the CONTROL state model.
- Whenever the ON-LINE/REMOTE state is active and the operator issues a command to the equipment, the equipment shall cause an ‘operator command issued’ event.

7.13.5 *Scenarios*

7.13.5.1 *Operator-Initiated Scenarios*

7.13.5.1.1.1 Host Accepts ON-LINE:

| COMMENTS | HOST | EQUIPMENT | COMMENTS |
|---|---|---|---|
| Operator actuates ON-LINE switch when equipment OFF-LINE state is active. | | | |
| | | <--S1,F1 | Equipment requests ON-LINE. |
| Host grants ON-LINE. | S1,F2--> | | |
| | | <--S6,F11 | “Control State LOCAL (or REMOTE)” event. |
| Acknowledge. | S6,F12--> | | |

[1] If there is no host response (i.e., reply timeout), the equipment shall treat it as a denial.

LETTER BALLOT

7.13.5.1.1.2 Host Denies ON-LINE:

| COMMENTS | HOST | EQUIPMENT | COMMENTS |
|---|---|---|---|
| Operator actuates ON-LINE switch when equipment OFF-LINE state is active. | | | |
| | | <--S1,F1 | Equipment requests ON-LINE. |
| Host denies ON-LINE. | S1,F0--> | | |

7.13.5.1.1.3 Operator Sets OFF-LINE:

| COMMENTS | HOST | EQUIPMENT | COMMENTS |
|---|---|---|---|
| Operator actuates OFF-LINE switch when equipment ON-LINE state is active. | | | |
| | | <--S6,F11 | "Equipment requests OFF-LINE" event. |
| Acknowledge. | S6,F12--> | | |

7.13.5.1.1.4 Operator Sets REMOTE:

| COMMENTS | HOST | EQUIPMENT | COMMENTS |
|---|---|---|---|
| Operator sets switch from LOCAL to REMOTE. | | | |
| | | <--S6,F11 | "Control State REMOTE" event. |
| Acknowledge. | S6,F12--> | | |

7.13.5.1.1.5 Operator Sets LOCAL:

| COMMENTS | HOST | EQUIPMENT | COMMENTS |
|---|---|---|---|
| Operator sets switch from REMOTE to LOCAL. | | | |
| | | <--S6,F11 | "Control State LOCAL" event. |
| Acknowledge. | S6,F12--> | | |

7.13.5.2 *Host-Initiated Scenarios*

7.13.5.2.1 Host Sets OFF-LINE:

| COMMENTS | HOST | EQUIPMENT | COMMENTS |
|---|---|---|---|
| Host requests OFF-LINE. | S1,F15--> | | |
| | | | [IF] Equipment is OFF-LINE [THEN]: |
| | | <--S1,F0 | Equipment does not process requests. |
| | | | [ELSE] Equipment ON-LINE |
| | | <--S1,F16 | Equipment acknowledges request and transitions to OFF-LINE. |
| | | <--S6,F11 | "Equipment OFF-LINE" event. |
| Acknowledge | S6,F12--> | | |
| | | | [END_IF] |

7.13.5.2.2 Host Sets ON-LINE:

| COMMENTS | HOST | EQUIPMENT | COMMENTS |
|---|---|---|---|
| Host requests ON-LINE. | S1,F17--> | | |
| | | | [IF] Equipment is HOST OFF-LINE state not active. [THEN] |
| | | <--S1,F18 | Equipment denies request (ONLACK = 1). |
| | | | [ELSE] Equipment HOST OFF-LINE state is active. |
| | | <--S1,F18 | Equipment acknowledges request (ONLACK != 1). |
| | | <--S6,F11 | "Control state LOCAL (or REMOTE)" event. (only if ONLACK = 0) |
| Acknowledge | S6,F12--> | | |
| | | | [END_IF] |

## 8 Data Items

8.1 The following sections specify which data items and variable data items are required.

8.2 Except for the specified format restrictions, all data items and variable data items follow the definitions contained in SEMI E5.

8.3 *Data Item Restrictions* — The following is a subset of the data items used by SECS-II messages specified in this Standard. Each data item listed in this section is restricted in its SEMI E5 defined usage. Most are limited to a single format from their standard list of formats. Data items used by SECS-II messages contained in this Document, but which have no restrictions are not duplicated here.

NOTE 20: One data item, ALCD, is restricted in other than its format. Take note of this restriction as described below.

NOTE 21: The equipment supplier shall document any restrictions on length or format of CPNAME. Suppliers shall document the behavior of spaces in a CPNAME. The maximum length of CPNAME shall be 40. This change became effective in September 1995.

| | |
|---|---|
| ACKC7A | Format: 51 |
| ALCD | Only bit 8 (alarm set/cleared) of the binary byte is used. Bits 1–7, denoting alarm category, are not used. |
| ALID | Format: 5() |
| CCODE | Format: 20, 32, 34, 52, 54 |
| CEID | Format: 5() |
| CPNAME | Format: 20 |
| DATAID | Format: 5() |
| DATALENGTH | Format: 5() |
| ECID | Format: 5() |
| LENGTH | Format: 5() |
| PPID | Format: 20 |
| RCMD | Format: 20 |
| REPGSZ | Format: 5() |
| RPTID | Format: 5() |
| SEQNUM | Format: 52 |
| SMPLN | Format: 5() |
| SVID | Format: 5() |
| TEXT | Format: 20 |
| TOTSMP | Format: 5() |
| TRID | Format: 5() |
| VID | Format: 5() |

8.4 *Variable Item List* — The following variable data items from the Variable Item Dictionary in SEMI E5 are required. Format restrictions are noted.

8.4.1 DVVAL's:

AlarmID Format: 5()
EventLimit
LimitVariable
PPChangeName Format: 20 — required only if process programs are implemented
PPChangeStatus — required only if process programs are implemented
PPError — required only if process programs are implemented
RcpChangeName — required only if E42 recipes are implemented
RcpChangeStatus — required only if E42 recipes are implemented
TransitionType

8.4.2 ECV's:

EnableSpooling
EstablishCommunicationsTimeout
MaxSpoolTransmit
OverWriteSpool
TimeFormat

8.4.3 SV's:

AlarmsEnabled
AlarmsSet
Clock
ControlState
EventsEnabled

PPError — required only if process programs are implemented
PPExecName Format: 0,20 — required only if process programs are implemented
PPFormat — required only if process programs or E42 recipes are implemented
PreviousProcessState
ProcessState
RcpExecName — required only if E42 recipes or E139 recipes are implemented
SpoolCountActual
SpoolCountTotal
SpoolFullTime
SpoolStartTime

## 9 Collection Events

9.1 Table 8 provides the list of collection events required to support the capabilities addressed within this Standard. Also shown are typical variable data that would most likely be included in the associated collection event report and a reference to the event trigger and to the appropriate section of the Standard.

9.2 This list does not represent all events that might be needed to properly monitor/control equipment. Many events are unique to the specific equipment characteristics. The needed additions are a matter for other standards and for collaboration between equipment supplier and user.

9.3 See § 6.4 for further detail of variable data items.

**Table 8 GEM-Defined Collection Events**

| *Event Designation* | *Typical Variable Data* | *Reference* |
|---|---|---|
| Control-Related Events: | | § 4.5 |
| Equipment OFF-LINE | ControlState, Clock | ON-LINE->OFF-LINE |
| Control State LOCAL | ControlState, Clock | REMOTE->LOCAL or OFF-LINE->LOCAL |
| Control State REMOTE | ControlState, Clock | LOCAL->REMOTE or OFF-LINE->REMOTE |
| Operator Command Issued | OperatorCommand | Operator Activity while REMOTE state is active. |
| Processing-Related Events: | See [#1] | § 4.6 |
| Processing Started | Clock, PreviousProcessState | Entry into EXECUTING state. |
| Processing Completed | Clock, PreviousProcessState | Normal exit of EXECUTING state. |
| Processing Stopped | Clock, PreviousProcessState | Result of STOP command from host or operator. |
| Processing State Change | Clock, ProcessState, PreviousProcessState | Any processing state transition. |
| Alarm Management Events: | | § 5.4 |
| Alarm$_n$Detected | Clock, AlarmID, AlarmsSet, Associated variable data | ALARM$_n$CLEAR->ALARM$_n$SET |
| Alarm$_n$Cleared | Clock, AlarmID, AlarmsSet | ALARM$_n$SET->ALARM$_n$CLEAR |
| Equipment Constant Events: | | § 5.6 |
| Operator Equipment Constant Change | ECID | Operator activity |
| Limits Monitoring: | | § 5.3.4 |
| Limit Zone Transition$_n$ (separate CEID per variable) | Clock, LimitVariable, EventLimit, Transition Type | Entry into BELOW LIMIT or ABOVE LIMIT states. |
| Process Program Management Events: | | § 5.7 |
| Process Program Change (required only if process programs are implemented) | PPChangeName, PPChangeStatus | Operator activity |

| *Event Designation* | *Typical Variable Data* | *Reference* |
|---|---|---|
| Process Recipe (s) Selected | PPExecName (required only if process programs are implemented)<br>RcpExecName (required only if E42 recipes or E139 recipes are implemented) | Operator/Host activity |
| Material Movement Events: | | § 5.8 |
| Material Received | Clock | |
| Material Removed | Clock | |
| Spooling Events: | | § 5.12 |
| Spooling Activated | SpoolStartTime | SPOOL INACTIVE->SPOOL ACTIVE |
| Spooling Deactivated | SpoolCountTotal | SPOOL OUTPUT->SPOOL INACTIVE |
| Spool Transmit Failure | Clock, SpoolCountActual SpoolCountTotal | TRANSMIT SPOOL->NO SPOOL OUTPUT |
| Terminal Services Events: | | § 5.9 |
| Message Recognition | Clock | Operator |
| New Execution Recipe Event (required only if E42 recipes are implemented) | RcpChangeName, RcpChangeStatus | § 5.7.2.2 |
| Execution Recipe Change Event (required only if E42 recipes are implemented) | RcpChangeName, RcpChangeStatus | § 5.7.2.2 |
| Successful Upload (required only if large E42 recipes or large process programs are implemented) | DataSetName | § 5.7.5.3 and § 5.7.6.3 |
| Bad Upload (required only if large E42 recipes or large process programs are implemented) | DataSetName | § 5.7.5.3 and § 5.7.6.3 |

#1 Any transition in the implemented processing state model must have a corresponding collection event.

## 10 SECS-II Message Subset

10.1 This section lists the required set of SECS-II messages as referenced in this Document. Definitions for these messages can be found in SEMI E5. All primary messages (for which SEMI E5 defines replies) should have replies available. Replies are required or optional as specified in SEMI E5.

**STREAM 1: Equipment Status**

| | |
|---|---|
| S1,F1 Are You There Request (R) | S,H<->E |
| S1,F2 On-Line Data (D) | S,H<->E |
| S1,F3 Selected Equipment Status Request (SSR) | S,H->E |
| S1,F4 Selected Equipment Status Data (SSD) | M,H<-E |
| S1,F11 Status Variable Namelist Request (SVNR) | S,H->E |
| S1,F12 Status Variable Namelist Reply (SVNRR) | M,H<-E |
| S1,F13 Establish Communications Request (CR) | S,H<->E |
| S1,F14 Establish Communications Request Acknowledge (CRA) | S,H->E,reply |
| S1,F15 Request OFF-LINE (ROFL) | S,H<-E |
| S1,F16 OFF-LINE Acknowledge (OFLA) | S,H->E,reply |
| S1,F17 Request ON-LINE (RONL) | S,H<-E |
| S1,F18 ON-LINE Acknowledge (ONLA) | S,H<-E |
| S1,F21 Data Variable Namelist Request (DVNR) | S,H->E, reply |

| | |
|---|---|
| S1,F22 Data Variable Namelist (DVN) | M,H<-E |
| S1,F23 Collection Event Namelist Request (CENR) | S,H->E, reply |
| S1,F24 Collection Event Namelist (CEN) | M,H<-E |

**STREAM 2: Equipment Control and Diagnostics**

| | |
|---|---|
| S2,F13 Equipment Constant Request (ECR) | S,H->E |
| S2,F14 Equipment Constant Data (ECD) | M,H<-E |
| S2,F15 New Equipment Constant Send (ECS) | S,H->E |
| S2,F16 New Equipment Constant Acknowledge (ECA) | S,H<-E |
| S2,F17 Date and Time Request (DTR) | S,H<->E |
| S2,F18 Date and Time Data (DTD) | S,H<->E |
| S2,F23 Trace Initialize Send (TIS) | S,H->E |
| S2,F24 Trace Initialize Acknowledge (TIA) | S,H<-E |
| S2,F29 Equipment Constant Namelist Request (ECNR) | S,H->E |
| S2,F30 Equipment Constant Namelist (ECN) | M,H<-E |
| S2,F31 Date and Time Send (DTS) | S,H->E |
| S2,F32 Date and Time Acknowledge (DTA) | S,H<-E |
| S2,F33 Define Report (DR) | M,H->E |
| S2,F34 Define-Report Acknowledge (DRA) | S,H<-E |
| S2,F35 Link Event Report (LER) | M,H->E |
| S2,F36 Link Event Report Acknowledge (LERA) | S,H<-E |
| S2,F37 Enable/Disable Event Report (EDER) | S,H->E,reply |
| S2,F38 Enable/Disable Event Report Acknowledge (EDEA) | S,H<-E |
| S2,F39 Multi-Block Inquire (DMBI) | S,H->E |
| S2,F40 Multi-Block Grant (DMBG) | S,H<-E |
| S2,F41 Host Command Send (HCS) | S,H->E |
| S2,F42 Host Command Acknowledge (HCA) | S,H<-E |
| S2,F43 Reset Spooling Streams and Functions (RSSF) | S,H->E |
| S2,F44 Reset Spooling Acknowledge (RSA) | M,H<-E |
| S2,F45 Define Variable Limit Attributes (DVLA) | M,H->E |
| S2,F46 Variable Limit Attribute Acknowledge (VLAA) | M,H<-E |
| S2,F47 Variable Limit Attribute Request (VLAR) | S,H->E |
| S2,F48 Variable Limit Attributes Send (VLAS) | M,H<-E |
| S2,F49 Enhanced Remote Command | M,H->E |
| S2,F50 Enhanced Remote Command Acknowledge | M,H<-E |

**STREAM 5: Exception (Alarm) Reporting**

| | |
|---|---|
| S5,F1 Alarm Report Send (ARS) | S,H<-E |
| S5,F2 Alarm Report Acknowledge (ARA) | S,H->E |
| S5,F3 Enable/Disable Alarm Send (EAS) | S,H->E |
| S5,F4 Enable/Disable Alarm Acknowledge (EAA) | S,H<-E |
| S5,F5 List Alarms Request (LAR) | S,H->E |
| S5,F6 List Alarm Data (LAD) | M,H<-E |

LETTER BALLOT

**STREAM 6: Data Collection**

| | |
|---|---|
| S6,F1 Trace Data Send (TDS) | S,H<-E |
| S6,F2 Trace Data Acknowledge (TDA) | S,H->E |
| S6,F5 Multi-block Data Send Inquire (MBI) | S,H<-E |
| S6,F6 Multi-block Grant (MBG) | S,H->E |
| S6,F11 Event Report Send (ERS) | M,H<-E |
| S6,F12 Event Report Acknowledge (ERA) | S,H->E |
| S6,F15 Event Report Request (ERR) | S,H->E |
| S6,F16 Event Report Data (ERD) | M,H<-E |
| S6,F19 Individual Report Request (IRR) | S,H->E |
| S6,F20 Individual Report Data (IRD) | M,H<-E |
| S6,F23 Request Spooled Data (RSD) | S,H->E |
| S6,F24 Request Spooled Data Acknowledgement Send (RSDAS) | S,H<-E |

**STREAM 7: Process Program Load – Required only if equipment implements process programs**

| | |
|---|---|
| S7,F1 Process Program Load Inquire (PPI) | S,H<->E,reply |
| S7,F2 Process Program Load Grant (PPG) | S,H<->E |
| S7,F3 Process Program Send (PPS) | M,H<->E |
| S7,F4 Process Program Acknowledge (PPA) | S,H<->E |
| S7,F5 Process Program Request (PPR) | S,H<->E |
| S7,F6 Process Program Data (PPD) | M,H<->E |
| S7,F17 Delete Process Program Send (DPS) | S,H->E |
| S7,F18 Delete Process Program Acknowledge (DPA) | S,H<-E |
| S7,F19 Current EPPD Request (RER) | S,H->E |
| S7,F20 Current EPPD Data (RED) | M,H<-E |
| S7,F23 Formatted Process Program Send (FPS) | M,H<->E |
| S7,F24 Formatted Process Program Acknowledge (FPA) | S,H<->E |
| S7,F25 Formatted Process Program Request (FPR) | S,H<->E |
| S7,F26 Formatted Process Program Data (FPD) | M,H<->E |
| S7,F27 Process Program Verification Send (PVS) | S,H<-E |
| S7,F28 Process Program Verification Acknowledge (PVA) | S,H->E |
| S7,F29 Process Program Verification Inquire (PVA) | |
| S7,F30 Process Program Verification Grant (PVG) | S,H<->E,reply |
| S7,F37 Large Process Program Send | S,H<->E |
| S7,F38 Large Process Program Acknowledge | S,H<->E,reply |
| S7,F39 Large Formatted Process Program Send | S,H<->E |
| S7,F40 Large Formatted Process Program Acknowledge | S,H<->E,reply |
| S7,F41 Large Process Program Request | S,H<->E |
| S7,F42 Large Process Program Acknowledge | |
| S7,F43 Large Formatted Process Program Request | S,H<->E,reply |
| S7,F44 Large Formatted Process Program Acknowledge | S,H<->E |

LETTER BALLOT

**STREAM 9: System Errors**

| | |
|---|---|
| S9,F1 Unrecognized Device ID (UDN) | S,H<-E |
| S9,F3 Unrecognized Stream Type (USN) | S,H<-E |
| S9,F5 Unrecognized Function Type (UFN) | S,H<-E |
| S9,F7 Illegal Data (IDN) | S,H<-E |
| S9,F9 Transaction Timer Timeout (TTN) | S,H<-E |
| S9,F11 Data Too Long (DLN) | S,H<-E |
| S9,F13 Conversation Timeout (CTN) | S,H<-E |

**STREAM 10: Terminal Services**

| | |
|---|---|
| S10,F1 Terminal Request (TRN) | S,H<-E |
| S10,F2 Terminal Request Acknowledge (TRA) | S,H->E |
| S10,F3 Terminal Display, Single (VTN) | S,H->E |
| S10,F4 Terminal Display, Single Acknowledge (VTA) | S,H<-E |
| S10,F5 Terminal Display, Multi-block (VMN) | M,H->E |
| S10,F6 Terminal Display, Multi-block Acknowledge (VMA) | S,H<-E |
| S10,F7 Multi-block Not Allowed (MNN) | S,H<-E |

**STREAM 13: – Required only if equipment implements E139 recipes, large E42 recipes or large process programs**

| | |
|---|---|
| S13,F1 Send Data Set Send (DSSS) | S,H<->E,reply |
| S13,F2 Send Data Set Acknowledge (DSSA) | S,H<->E |
| S13,F3 Open Data Set Request (DSOR) | S,H<->E,reply |
| S13,F4 Open Data Set Data (DSOD) | S,H<->E |
| S13,F5 Read Data Set Request (DSSR) | S,H<->E,reply |
| S13,F6 Read Data Set Data (DSRD) | M,H<->E |
| S13,F7 Close Data Set Send (DSCS) | S,H<->E,reply |
| S13,F8 Close Data Set Acknowledge (DSCA) | S,H<->E |
| S13,F9 Reset Data Set Send (DSRS) | S,H<->E,reply |
| S13,F10 Reset Data Set Acknowledge (DSRA) | S,H<->E |

**STREAM 14: Object Services**

| | |
|---|---|
| S14,F1 GetAttr Request | S,H<->E |
| S14,F2 GetAttr Data | M,H<->E |

**STREAM 15: Recipe Management – Required only if equipment implements E42 recipes**

| | |
|---|---|
| S15,F1 Recipe Management Multi-block Inquire | S,H<->E |
| S15,F2 Recipe Management Multi-block Grant | S,H<->E |
| S15,F21 Recipe Action Request | M,H<->E |
| S15,F22 Recipe Action Acknowledge | M,H->E |
| S15,F27 Recipe Download Request | M,H->E |
| S15,F28 Recipe Download Acknowledge | M,H<-E |
| S15,F29 Recipe Verify Request | M,H->E |
| S15,F30 Recipe Verify Data | M,H<-E |
| S15,F31 Recipe Upload Request | S,H->E |

| | |
|---|---|
| S15,F32 Recipe Upload Data | M,H<-E |
| S15,F35 Recipe Delete Request | M,H->E |
| S15,F36 Recipe Delete Acknowledge | M,H<-E |
| S15,F49 Large Recipe Download Request | S,H->E,reply |
| S15,F50 Large Recipe Download Acknowledge | S,H<-E |
| S15,F51 Large Recipe Upload Request | S,H->E,reply |
| S15,F52 Large Recipe Upload Acknowledge | S,H<-E |
| S15,F53 Recipe Verification Send | M,H<-E,reply |
| S15,F54 Recipe Verification Acknowledge | S,H->E |

**STREAM 19: Recipe and Parameter Management – Required only if E139 recipes are implemented**

| | |
|---|---|
| S19,F1 Get PDE Directory (GPD) | M,H<->E,reply |
| S19,F2 PDE Directory Data (PDD) | M,H<->E |
| S19,F3 Delete PDE (DPDE) | M,H->E,reply |
| S19,F4 Delete PDE Acknowledge (DPDEA) | M,H<-E |
| S19,F5 Get PDE Header (GPH) | M,H<->E,reply |
| S19,F6 PDE Header Data (PHD) | M,H<->E |
| S19,F7 Get PDE (GPDE) | M,H<->E,reply |
| S19,F8 PDE Data (PDED) | M,H<->E |
| S19,F9 Request To Send PDE (RTSP) | S,H<->E,reply |
| S19,F10 Send PDE Grant (SPDEG) | S,H<->E |
| S19,F11 Send PDE (SPDE) | S,H<->E,reply |
| S19,F12 Send PDE Acknowledge (SPDEA) | S,H<->E |
| S19,F13 TransferContainer Report (TR) | M,H<->E,reply |
| S19,F14 PDE TransferContainer Report Acknowledge (TA) | S,H<->E |
| S19,F15 Resolve PDE Request (RPR) | M,H->E,reply |
| S19, F16 Resolve PDE Data (RPD) | M,H<-E |
| S19,F17 Verify PDE (VP) | M,H->E,reply |
| S19,F18 Verify PDE Data (VPD) | M,H<-E |
| S19,F19 RaP Multi-block Inquire (RMI) | S,H<->E,reply |
| S19,F20 RaP Multi-block Grant (RMG) | S,H<->E |

## 11 GEM Compliance

11.1 This section defines compliance to the GEM Standard. It describes the fundamental GEM requirements and additional GEM capabilities. It provides references to other sections of the Standard where detailed requirements are located. This section also defines standard terminology and documentation that can be used by equipment suppliers and device manufacturers to describe compliance with this Standard.

11.1.1 The GEM Standard contains two types of specifications:

- fundamental GEM requirements and
- requirements pertaining to additional GEM capabilities.

11.1.2 The fundamental GEM requirements form the foundation of the GEM Standard. The additional GEM capabilities provide functionality required for some types of factory automation or functionality applicable to specific types of equipment. Figure 12 illustrates the relationship of the fundamental GEM requirements and the additional GEM capabilities.

11.2 *Fundamental GEM Requirements* — All equipment shall comply with the fundamental GEM requirements listed in Table 9. Compliance to these requirements involves precise and complete adherence to all sections of the GEM Standard referenced in Table 9.

**Table 9 Fundamental GEM Requirements**

| *Requirement* | *Section References* |
|---|---|
| State Models | 4, 4.3, 4.5 |
| Equipment Processing States | 4.6 |
| Host-Initiated S1,F13/F14 Scenario | 5.2.5.1 |
| Event Notification | 5.3.1.2 |
| On-line Identification | 5.3.6 |
| Error Messages | 5.10 |
| Control (Operator-Initiated) | 5.13 (except 5.13.5.2) |
| Documentation | 9.5 |

11.2.1 In addition, compliance requires adherence to the portions of the following sections that are applicable to the fundamental GEM requirements:

- Variable data items (GEM, § 6)
- SECS-II data item restrictions (GEM, § 6)
- Collection events (GEM, § 7)

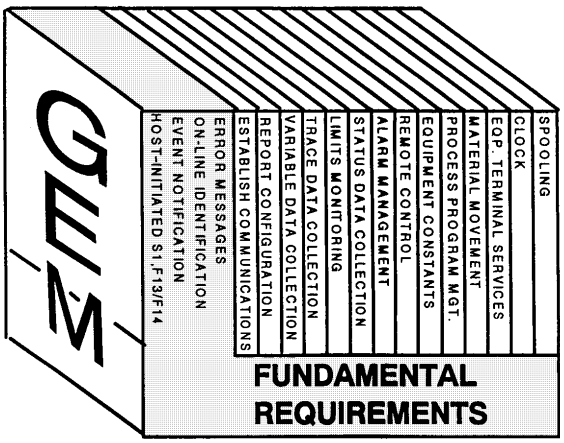



**Figure 13**
**GEM Requirements and Capabilities**

11.3 *GEM Capabilities* — The following table lists all GEM capabilities and the sections of the GEM Standard where they are specified. These sections contain the detailed requirements for implementing a GEM capability. Requirements for an individual capability include any referenced portions of the Document. As an example, the Alarm Management capability requires implementation of the status variables ‘AlarmsEnabled’ and ‘AlarmsSet’ as defined in § 6.

**Table 10 Section References for GEM Capabilities**

| *Capability* | *Section References* |
|---|---|
| Establish Communications | 5.2, 4.4 |
| Event Notification | 5.3.1.2 |
| Dynamic Event Report Configuration | 5.3.1.3 |
| Data Variable and Collection Event Namelist Requests | 5.3.1.4 |
| Variable Data Collection | 5.3.2 |
| Trace Data Collection | 5.3.3 |
| Limits Monitoring | 5.3.4 |
| Status Data Collection | 5.3.5 |
| On-line Identification | 5.3.6 |
| Alarm Management | 5.4 |
| Remote Control | 5.5 |
| Equipment Constants | 5.6 |
| Process Recipe Management | 5.7 |
| Material Movement | 5.8 |
| Equipment Terminal Services | 5.9 |
| Error Messages | 5.10 |
| Clock | 5.11 |
| Spooling | 5.12 |
| Control (Operator-Initiated) | 5.13 (except 5.13.5.1) |
| Control (Host-Initiated) | 5.13.5.1 |

11.4 *Definition of GEM Compliance* — The term ‘GEM Compliance’ is defined with respect to individual GEM capabilities to indicate adherence to the GEM Standard for a specific capability. Equipment is GEM-compliant for a specific GEM capability if, and only if, the following three criteria are met:

- The fundamental GEM requirements are satisfied.
- The capability is implemented to conform with all applicable definitions, descriptions, and requirements defined for the capability in this Standard.
- The equipment does not exhibit behavior related to this capability that conflicts with the GEM behavior defined for the capability.

11.4.1 For example, equipment that provides SECS-II messages for management of process programs must precisely implement the GEM Process Program Management capability to be ‘GEM-Compliant for Process Program Management.’

11.4.2 Equipment may supply additional functionality not specified in the GEM Standard by using any messages defined in the SECS-II Standard as long as the additional functionality does not conflict with compliance to GEM capabilities.

11.4.3 Figure 13 illustrates the host view of equipment communications in relationship to the components of the GEM Standard. The GEM capabilities are built upon the fundamental GEM requirements and present GEM-compliant behavior to the host when they are not obstructed by conflicting functionality. Additional non-GEM capabilities and non-obstructing extensions to GEM capabilities provide additional functionality while maintaining GEM behavior from the host view.

11.4.4 One additional term is defined to facilitate discussion of GEM capability. Equipment is ‘Fully GEM Capable’ if and only if it meets the following two criteria:

- The equipment supplies all the GEM capabilities listed in § 9.3.
- Every implemented GEM capability is GEM-Compliant.

11.5 *Documentation* — This section describes documentation requirements in addition to those specified in § 4 and § 5 of this Standard. All documentation of the SECS-II interface shall be supplied as a single volume, including Message Documentation, a Compliance Statement and the documentation required by § 4 and § 5.

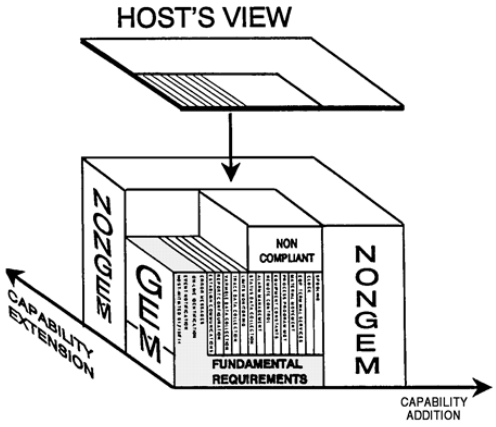



**Figure 14**
**Host View of GEM**

11.5.1 *Message Documentation* — The equipment supplier shall provide message documentation in conformance with § 11 (Message Documentation) of SEMI E5.

11.5.2 *GEM Compliance Statement* — The SECS-II interface documentation provided by an equipment supplier shall address GEM compliance. This documentation shall include a GEM Compliance Statement that accurately indicates for each capability whether it has been implemented and whether it has been implemented in a GEM-compliant manner. The format for this statement is supplied as Table 11.

11.5.2.1 The table consists of three columns. The first column lists the requirements and capabilities. The other two columns pose questions to the supplier:

- *Implemented* — Does the equipment provide functionality that is similar to that defined for the GEM requirement or capability?
- *GEM-Compliant* — Has that requirement or capability been implemented in a GEM-compliant manner?

11.5.3 The equipment supplier may provide documentation on the format of required data items (see§ 6) using SECS Message Language™ Notation (SML®). The SML formats are provided in Table 12.

**Table 11 GEM Compliance Statement**

| **GEM COMPLIANCE STATEMENT** | | |
|---|---|---|
| *FUNDAMENTAL GEM REQUIREMENTS* | *IMPLEMENTED* | *GEM-COMPLIANT* |
| State Models | □ Yes □ No | □ Yes[#1] □ No |
| Equipment Processing States | □ Yes □No | |
| Host-Initiated S1,F13/F14 Scenario | □Yes □No | |
| Event Notification | □ Yes □No | |
| On-Line Identification | □ Yes □ No | |

LETTER BALLOT

| *FUNDAMENTAL GEM REQUIREMENTS* | *IMPLEMENTED* | *GEM-COMPLIANT* |
|---|---|---|
| Error Messages | □ Yes □ No | □ Yes[#1] □ No |
| Documentation | □ Yes □ No | |
| Control (Operator Initiated) | □ Yes □ No | |
| *ADDITIONAL CAPABILITIES* | *IMPLEMENTED* | *GEM-COMPLIANT*[#2] |
| Establish Communications | □ Yes □ No | □ Yes □ No |
| Dynamic Event Report Configuration | □ Yes □ No | □ Yes □ No |
| Data Variable and Collection Event Namelist Requests | □ Yes □ No | □ Yes □ No |
| Variable Data Collection | □ Yes □ No | □ Yes □ No |
| Trace Data Collection | □ Yes □ No | □ Yes □ No |
| Status Data Collection | □ Yes □ No | □ Yes □ No |
| Alarm Management | □ Yes □ No | □ Yes □ No |
| Remote Control | □ Yes □ No | □ Yes □ No |
| Equipment Constants | □ Yes □ No | □ Yes □ No |
| Process Recipe Management | □ Yes □ No | Process Programs: □ Yes □ No<br>E42 Recipes: □ Yes □ No<br>E139 Recipes: □ Yes □ No |
| Material Movement | □ Yes □ No | □ Yes □ No |
| Equipment Terminal Services | □ Yes □ No | □ Yes □ No |
| Clock | □ Yes □ No | □ Yes □ No |
| Limits Monitoring | □ Yes □ No | □ Yes □ No |
| Spooling | □ Yes □ No | □ Yes □ No |
| Control (Host-Initiated) | □ Yes □ No | □ Yes □ No |

#1 Do not mark YES unless all fundamental GEM requirements are implemented and GEM-compliant.

#2 Additional capabilities may not be marked GEM-compliant unless the fundamental GEM requirements are GEM-compliant.

**Table 12 SML Notation**

| *Item Format* | *SECS-II Format Code* | | *SML Item Format Mnemonic* |
|---|---|---|---|
| | *Binary* | *Octal* | |
| LIST | 000000 | 00 | L [length] |
| Binary | 001000 | 10 | B |
| Boolean | 001001 | 11 | BOOLEAN |
| ASCII | 010000 | 20 | A [length] or A [min., max.] |
| JIS-8 | 010001 | 21 | J [length] or J [min., max.] |
| 8-byte integer (signed) | 011000 | 30 | I8 |
| 1-byte integer (signed) | 011001 | 31 | I1 |
| 2-byte integer (signed) | 011010 | 32 | I2 |
| 4-byte integer (signed) | 011100 | 34 | I4 |
| 8-byte floating point | 100000 | 40 | F8 |
| 4-byte floating point | 100100 | 44 | F4 |
| 8-byte integer (unsigned) | 101000 | 50 | U8 |
| 1-byte integer (unsigned) | 101001 | 51 | U1 |
| 2-byte integer (unsigned) | 101010 | 52 | U2 |
| 4-byte integer (unsigned) | 101100 | 54 | U4 |

LETTER BALLOT

## RELATED INFORMATION

LETTER BALLOT

## ~~A. Application Notes~~ RELATED INFORMATION 1

~~**NOTICE:** The material contained in these Application Notes is not an official part of this SEMI Standard and is not intended to modify or supersede the official Standard. Rather, these notes are auxiliary information describing possible methods for implementing the protocol described by the Standard and are included as reference material. The Standard should be referred to in all cases. SEMI makes no warranties or representations as to the suitability of the material set forth herein for any particular application. The determination of the suitability of the material is solely the responsibility of the user.~~

**NOTICE:** This Related Information is not an official part of SEMI E30 and was derived from the work of the Information and Control Global Technical Committee. This Related Information was approved for publication by full letter ballot procedures on [A&R approval date TBD].

### ~~A.1~~ R1-1 Factory Operational Script

An Operational Script is a series of capabilities arranged in a typical factory operation sequence. The intent of having an Operational Script is to help put the SECS-II message Scenarios into a context. Although this context will vary, it represents a typical operational sequence found in most semiconductor device manufacturers' applications.

- System Initialization
- Synchronization
- Machine Setup
- Production Setup
- Processing
- Post-Processing
- Shutdown

The following script is not intended to be complete, but to serve as an example to be further developed on an implementation basis.

~~A1~~R1-1.1 *Anytime Capabilities* — All capabilities can generally occur at any time during the operational script sequence.

~~A1~~R1-1.2 *System Initialization and Synchronization* — Upon system initialization, the default setting for communication (enabled or disabled) becomes effective, as well as any equipment constants or other information retained in non-volatile storage. The initial communication status is displayed at the equipment.

Assuming the communication state is enabled, the equipment will attempt to establish communication with a host computer. See § 5.2 for a description of the scenario for establishing communications.

Upon receiving an indication that the equipment was previously not communicating, the host would typically perform synchronization activities including setting the equipment's clock and requesting selected status information. Note that synchronization activity is host application-dependent and may be implemented using various scenarios.

~~A1~~R1-1.3 *Production Set-Up* — The host typically has the following information:

- what material
- what process step
- what process program to use (PPID)
- current equipment status, VID's, SVID's
- data collection requirements (trace data and event data)
- VID's needed

LETTER BALLOT

- Equipment constants (ECID's)

Based upon the above information, the host will perform setup activities as required. It must be verified that the correct process program is available and selected at the equipment.

~~A1~~R1-1.3.1 *Auxiliary Material and Manual Set-Up* — Auxiliary material can be checked and verified at this point. If status variables exist for auxiliary material, they may be requested by the host.

Any other manual, nonprocess, and/or nonproduct specific set-up also may take place at this point. The operator may interact with the equipment and the host. If the operator interacts with the equipment, the equipment communications link with the host should stay operational.

The operator and the host may exchange information via equipment terminal services.

~~A1~~R1-1.3.2 *Product/Process Set-Up* — Specific product and/or process information is communicated to the equipment prior to processing material.

~~A1~~R1-1.3.3 *Material Load* — The host may instruct an operator or a material handling system to deliver material to the equipment.

Once the material has arrived at the equipment, the equipment or the operator will notify the host.

~~A1~~R1-1.3.4 *Production Data Collection Set-Up* — The host instructs the equipment to collect event-based data. Reports are defined and linked to events. Event reports can be enabled or disabled.

The host instructs the equipment to collect data from the equipment based on time intervals.

The host configures the equipment to monitor specific variables and to send event reports when variables transition between monitoring zones.

~~A1~~R1-1.4 *Processing*

~~A1~~R1-1.4.1 *Start Process Executing* — The host or operator issues a command to start.

~~A1~~R1-1.4.2 *Equipment Signals End of Run* — When process execution is completed, the equipment generates events. If any of the events are enabled, they will be sent as event reports.

~~A1~~R1-1.5 *Post-Processing* — The equipment has completed processing material. It now makes the material available to the operator or material handling system for removal. The equipment signals the host that it is available for more work.

~~A1~~R1-1.5.1 *Material Unload* — Material is unloaded from the equipment by an operator or material handling system.

## ~~A.2~~ R1-2 Equipment Front Panel

In the GEM Standard, several requirements are stated that involve the display or input of information at the equipment front panel. The 'equipment front panel' refers to an area on the equipment that is available to the operator under normal use (i.e., without removing maintenance access panels). This may include a CRT display, keyboard, switches, and lights.

This application note provides some guidance for implementation of the GEM front panel capabilities. All of these requirements map directly to state models and capabilities defined in § 4 and § 5. All capabilities may be implemented in either hardware (buttons, switches, lights) or in a software/CRT equivalent.

~~A2~~R1-2.1 *Displays and Indicators* — The intent of various displays is to inform the operator of either the current state of the equipment or of a recent change of state (or both). Therefore, it is most useful if these displays are continuously visible and easily recognized at a distance. Required displays/indicators include:

*Communications State* — This means that three distinct states must be represented: DISABLED, ENABLED/NOT COMMUNICATING, and ENABLED/COMMUNICATING.

*Terminal Services* — An 'New Host Message' indicator must be supplied.

~~A2~~R1-2.2 *Switches/Buttons* — Note that discrete switches also contain information for the user. However, these tend to represent the desired states of the operator/user. The equipment's response to a change of a switch may not be instantaneous. Still, the current position of switches should be available to the operator.

It may be appropriate to limit the access to some switches and buttons. This might be done via any of the standard methods, keys, passwords, combinations, etc. This is especially true for system default switches that would not often be changed. Required switches/buttons include:

*Communications State System Default* — In what communications state should the equipment be when system initialization is complete? The choices are DISABLED and ENABLED.

*Communications State Selector* — This is a toggle or button that will initiate a transition from ENABLED to DISABLED or vice versa.

*Message Recognition Button* — This button is used to initiate an event message to the host which indicates that the 'New Host Message' has been read. This button should function only when the New Host Message Indicator is activated and when the received message is displayed in the terminal display.

## ~~A.3~~ R1-3 Examples of Equipment Alarms

Table ~~A.3~~ R1-1 provides alarm examples pertaining to various configurational aspects of equipment.

NOTE 22: It is important to stress that these are just examples intended to illustrate that alarms pertain to situations in which there exists a potential for exceeding physical safety limits associated with people, equipment, and material being processed as per the GEM definition of an alarm.

NOTE 23: The alarm capability is intended as an addition to standard safety alarms (e.g., lights, horns). There is no intent to replace direct operator reaction to such problems. Nor is there the expectation that the host can necessarily prevent or directly address such alarms.

An actual machine shall have an associated set of alarms defined by the manufacturer that pertains to its specific configuration and design. The equipment manufacturer is responsible for supplying documentation associated with these alarm definitions.

**Table ~~A.3~~ R1-1 Alarm Examples per Equipment Configuration**

| *Subsystem* | *Alarm Description* | *ALID* | *Trigger* | *Reset* | *Operator* | *Equipment* | *Material* |
|---|---|---|---|---|---|---|---|
| Mainframe Power Supply | Overvoltage | | Voltage supply over maximum limit | | | X | |
| | Undervoltage | | Voltage supply under minimum limit | | | X | |
| Internal Power Distribution Bus | AC Low | | AC under minimum limit | | | X | X |
| Cooling System | Overtemp | | Temperature over maximum | | X | | X |
| | Pressure Low | | Pressure below minimum | | | | X |

- *Subsystem* — The subsystem of the equipment to which the alarm is related.
- *Alarm Description* — Description of the alarm.
- *ALID* — The Alarm ID as specified by SECS-II.
- *Trigger* — Text description of what caused the alarm.
- *Reset* — Description of how to resolve the alarm condition.
- *Affected* — Who or what is affected by the alarm trigger: Operator, Equipment, and Material.

## ~~A.4~~ R1-4 Trace Data Collection Example

This example shows an implementation of the Trace Data Collection capability defined in § 5.3.3.

S2,F23 sent by host:

```
TRID = ABCD
DSPER = 000100 (One minute per period)
TOTSMP = 9
REPGSZ = 3
SVID1 = Temperature
SVID2 = Relative humidity
```

S6,F1 looks like this (starting at time 1 a.m.):

1st transmission <L,4>

```
1. ABCD (trace ID)
2. 3 (last sample of the transmission)
3.    88    5    01    01    03    00
     Year Month Day  Hour Min  Sec
4. <L, n> n = 2 SVID 's x REPGSZ of
   3 = 2 x 3 = 6
   72 (temperature)
   0.29 (relative humidity)
   73 (temp.)
   0.30 (r.h.)
   71 (temp.)
   0.30 (r.h.)
```

2nd transmission <L,4>

```
1. ABCD
2. 6
3. 88  05  01  01  06  00
                hr  min
4. <L,6>
      73
      0.31
      71
      0.32
      71
      0.31
```

3rd and last transmission <L,4>

```
1. ABCD
2. 9
3. 88  05  01  01  09  00
                hr  min
4. <L,6>
   71
   0.30
   72
   0.30
   71
   0.31
```

### ~~A.5~~ R1-5 Harel Notation

Harel's statecharts extend traditional state-transition diagrams with several additional concepts, most important of which are hierarchy and concurrence. Statecharts depict the behavior of a system by showing states it may take, events that prompt a change of state, and the composition of states. What follows is a very brief description of the symbols defined for use and how these are useful to describe a system. See Figure A.5.1 for the basic notational symbols.

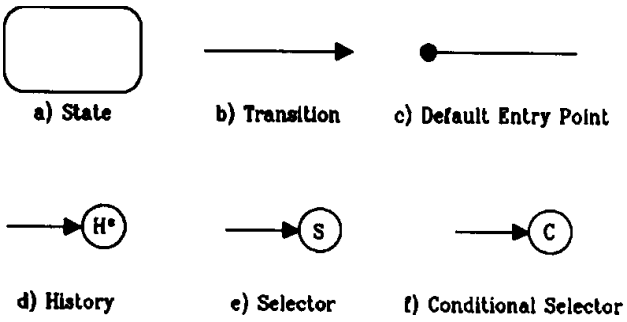



**Figure A.5.1**
**Harel Statechart Symbols**

States are represented by rounded boxes. A state transition is shown graphically with a line from the old state terminating with the arrow symbol at the new state. Transitions are unidirectional-while the reverse transition may be possible, it is considered a different transition with different conditions for initiation and different resultant actions.

States may be subdivided into substates to facilitate more concise definition of behavior. Thus, a hierarchy is defined whereby any state may be a substate of some parent state and in turn be the parent of its own substates. Substates must be one of two types, termed AND substates and OR substates.

A parent maybe divided into two or more OR substates of which one and only one is the active substate at any time. The accepted term for this exclusivity is XOR. Figure A.5.2 gives an example of a simple case of OR substates. In this example, some system (perhaps a motor) has a state named FUNCTIONAL. When the motor is FUNCTIONAL, it may be either ON or OFF, but never both.

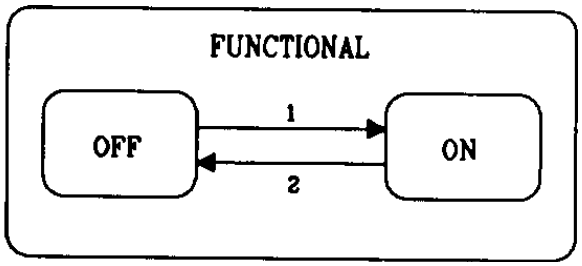



**Figure A.5.2**
**Example of OR Substates**

Another way of dividing a parent state corresponds roughly to subsystems. These AND substates represent parallelism, such that every AND substate of an active parent state is considered active. Harel also uses the term 'Orthogonal Component' to refer to AND substates. However, these parallel substates tend to be highly interactive and interdependent. For this reason, the word orthogonal is considered confusing and has been excluded from use in this document. Figure A.5.3 shows an example of AND substates representing (in part) an automobile. Note the convention of attaching the name of the parent state AUTOMOBILE to the outside of the state in a small box. The substates shown are independent components and may have their own substates (of either the AND or OR type):

- LIGHTS may be ON or OFF;
- DOOR may be OPEN or CLOSED;
- ENGINE is constructed of components such as pumps, pistons, carburetor, etc.

Exiting one of a set of AND substates requires the exit of all others. In some cases, a transition arrow will be shown from only one of the substates with the others implied.

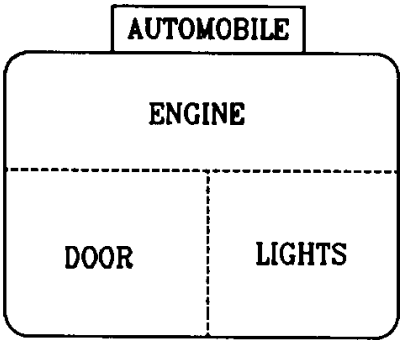



**Figure A.5.3**
**Example of AND Substates**

A simplification that also helps to prevent in determinacy is implemented with the symbol for default entry point. This symbol will indicate which OR substate is initially active when there is not an explicit choice. This lack of specification is indicated by a transition arrow from one state to another that does not cross the boundary of the parent to point specifically to a substate.

An entrance to a state terminating in a history symbol (see Figure A.5.1) indicates that the OR substate to be entered should be that which was active the last time the parent state was active (i.e., last time the car was running, the radio was on). The history symbol H refers to the choice of substates of the parent. The symbol H* extends further to the lowest level substates defined. In the absence of memory of a 'last time', the default entry is used.

The selector and conditional selector symbols serve to abbreviate complex entrances to states. Their meaning is similar and indicate that the choice of OR substate upon entry of a parent state depends on some condition that is not shown. The selector is usually used to combine several similar transition events, while the conditional selector will typically require some computation or test of conditions external to the stimulus for state transition. Please examine the referenced article for more detail.

NOTE 24: Within the body of this Document, the term statechart is not used in favor of the more traditional term state diagram.

~~A5~~R1-5.1 *State Definitions* — The state diagram provides a concise description of the function of a system. However, a full definition requires detail that cannot be included on the diagram. A description of each state is required that covers the boundaries of the state and any responses that occur within that state to the environment. The convention in this Document is to provide state names in ALL CAPS to help the reader identify where these are used. A sample state description of the ON state depicted in the Figure A.5.2 might be:

*ON*

The switch is in the on position. Power is available to the motor. Speed of the motor will change in proportion to the speed knob adjustment.

~~A5~~R1-5.2 *Transition Table* — The last piece of the state model is the transition table. It consists of several columns that list the transition number from the diagram, the starting and ending state for the transition, and three columns titled trigger, action, and comment. The trigger column describes the combination of events and conditions that initiates the transition (e.g., message Sx,Fy received). The trigger should be related to a single clearly defined event at the equipment. The action column identifies the activities associated directly with the transition. These activities may be of three types: (a) actions taken upon exit of the old state, (b) actions taken upon entry to the new state, and (c) actions not associated with either state. These are not differentiated in this Document. The final column allows for additional comments that help to clarify the transition. Table A.5, an example of transition table, illustrates the motor example in Figure A.5.2.

LETTER BALLOT

**Table ~~A.5~~ R1-5.1 Transition Table for Motor Example**

| # | *Current State* | *Trigger* | *New State* | *Action* | *Comment* |
|---|---|---|---|---|---|
| 1 | OFF | Switch turned to on position. | ON | Power supplied to motor. | Power supply assumed available. Motor begins to turn. |
| 2 | ON | Switch turned to off position. | OFF | Power supply to motor disconnected. | Motor begins deceleration. |

## ~~A.6~~ R1-6 Example Control Model Application

This section provides one example of a host's interaction with an equipment's control model. A host system must have a view of the control model to understand and predict equipment behavior. However, the implementor may simplify the host's view by assuming that some configuration settings are fixed and that the host-initiated features are not implemented. Applying these assumptions simplifies the behavior the host expects to see.

Figure A.6.1 shows the effective control model[1] based on the following host assumptions:

- The fundamental requirements are met, but the additional host-intiated control capability is not implemented.
- The configuration for the default entry to CONTROL is set to an OFF-LINE substate (either ATTEMPT ON-LINE or EQUIPMENT OFF-LINE).
- The destination state for transition 4 (failure of S1,F1 transaction) is configured to EQUIPMENT OFF-LINE.

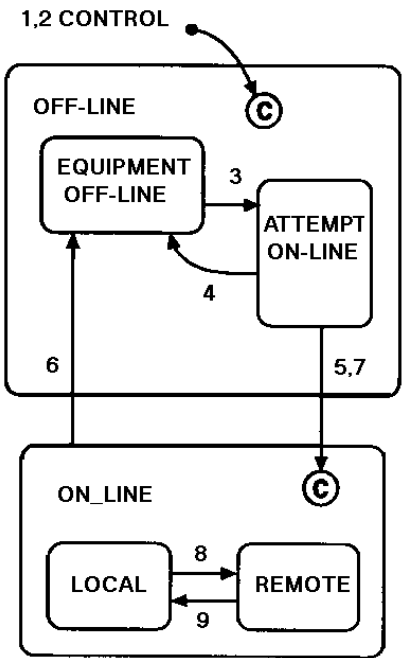



**Figure A.6.1**
**Example of the Simplified 'Effective' Control Model**

This view of the model has two further settings that the host recognizes as changeable at the equipment. The first is the configuration of which substate of OFF-LINE to be activated upon system initialization. The second is the front panel switch that determines whether the active system substate is LOCAL or REMOTE when ON-LINE.

[1] See § 4.5 for details of the control model.

This application has the following implications:

- This application requires that the equipment begin with the OFF-LINE state active. Thus, an equipment initiated S1,F1/F2 transaction must be completed before the equipment will begin sending all messages to the host.
- If a failed attempt to go ON-LINE is made by the equipment, it will not allow the host to complete the transition at a later time. An operator will be required to re-initiate the transition to ON-LINE when the host becomes ready.
- Once ON-LINE, the equipment will remain ON-LINE until an operator sets the equipment OFF-LINE at the equipment front panel.
- Since all transitions into the HOST OFF-LINE state are eliminated, this state is effectively eliminated from the host view of the control model.

This application retains the following features:

- The ON-LINE state is achieved only after the host acknowledges the equipment by replying to the S1,F1 with and S1,F2. This confirms to the operator attempting to put the equipment ON-LINE that the host application is ready for work to begin.
- It provides the operator the means to set the equipment OFF-LINE for non-host-related activities[1] (e.g., maintenance, test lots).
- The operator has the ability to operate the equipment with either the REMOTE or LOCAL state active. As the equipment transitions to ON-LINE, the preferred substate is automatically chosen (based on a front panel switch).
- The user may configure which substate of OFF-LINE the equipment will initially activate at system initialization. If ATTEMPT ON-LINE is chosen, the equipment will automatically attempt the transition to the ON-LINE state as system initialization.

## ~~A.7~~ R1-7 Examples of Limits Monitoring

~~A7~~R1-7.1 *Introduction*

~~A7~~R1-7.1.1 Four limits monitoring examples are included below to help clarify the use of limits and to illustrate typical applications. The first example shows how to apply limits to boolean values. The second illustrates application of several limits to a floating point variable in a classical control zone style. The third example shows an integer counter variable used to prompt for equipment maintenance.

~~A7~~R1-7.2 *Examples*

~~A7~~R1-7.2.1 *Example 1 — Valve Monitoring*

~~A7~~R1-7.2.1.1 The ACME Shine-Um-Rite Model 13 includes a sump which contains the chemical agent used to clean bare wafers. A chemical feeder system serves to refill the sump when the level drops below a certain level. The fill is accomplished via an on-off value driven by sensors in the sump. Facilities must be informed of the proportion of the time the valve is open (approximates usage) and any time the value remains open for more than 5 minutes (valve likely broken).

~~A7~~R1-7.2.1.2 To implement this requirement, a limit was defined for the Boolean status variable which contains the current state of the valve (i.e., 0 = Closed, 1 = Open). See Figure A7.1 for illustration. LIMITID1 was defined with UPPERDB = LIMITMAX = 1 (Open) and LOWERDB = LIMITMIN = 0 (Closed). As a result, any time the valve opens, a collection event is generated with TransitionType = 0 and when the valve closes, a collection event is generated with TransitionType = 1. An event report containing the DVVAL LimitVariable was attached to each collection event and reporting for the event was enabled.

NOTE 25: Boolean values are defined as 0 = False/Closed/Off and any value > 0 = True/Open/On–never depend on a value of 1.

[1] Which activities are ‘non-host-related’ varies from factory to factory. In general, fewer activities are ‘non-host-related’ as a factory’s automation level increases.

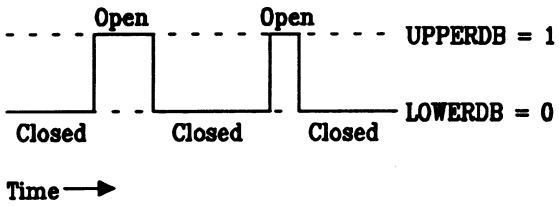



**Figure A7.1**
**Valve Monitoring Example**

~~A7~~R1-7.2.2 *Example 2 — Environment Monitoring*

~~A7~~R1-7.2.2.1 ACME also makes a Model 2 Stepper. The environmental control system of this equipment is designed to hold the internal temperature relatively constant, but is sensitive to large changes in the external environment, opening of access doors, etc. To ensure that processing conditions are appropriate, the internal stepper temperature is monitored to ensure it remains in a safe operating zone (within 'Shutdown' limits). In addition, a second set of limits are used within the Shutdown limits to bound the 'Normal' operating range. Frequent excursions from the normal range into the 'warning' range will prompt service on the environmental control system. The target temperature range is specified as 98° to 100°, the shutdown limits as 95° to 103°.

~~A7~~R1-7.2.2.2 Event reports are desired when the internal temperature moves outside of the normal operating zone into a warning zone (above or below), when the temperature moves back into the normal operating zone from the warning zones, and when the temperature moves out of the warning zones into the shutdown zones. Furthermore, temperature fluctuations of 0.5° should not trigger multiple event reports.

~~A7~~R1-7.2.2.3 Probably the most intuitive use of the limits monitoring capability is in establishing normal, warning, and shutdown zones for a particular equipment variable. Limits may be combined to provide such a scenario. The method is described below and illustrated in Figure A7.2. Please note that in the figure, limits are denoted as solid lines for simplicity, with deadbands indicated using the ± notation.

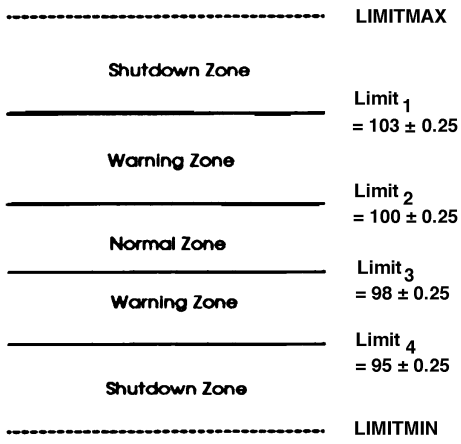



**Figure A7.2**
**Environment Monitoring Example**

~~A7~~R1-7.2.3 *Example 3 — Calibration Counter*

~~A7~~R1-7.2.3.1 Another ACME equipment is the multi-chamber Duz-It-All Model 7. This machine includes redundant chambers to increase throughput. One particular chamber on this equipment requires periodic calibration. The need for calibration is a nonlinear function of the number of wafers processed in that chamber. A status variable exists which contains the number of wafers processed since the last calibration was performed. Maintenance is definitely required after every 8 cycles, but the machine must be checked after 5 and 7 cycles to determine whether early calibration is necessary. This checking may be done by examining certain other equipment status variables.

~~A7~~R1-7.2.3.2 To meet this need, three limits are defined for the counter variable. Three limits were set, at 5, 7, and 8. Deadbands are set to zero, since chattering is not a problem. All the pertinent information is placed in an event report which is attached to the CEID for the limits of the counter to negate the need for further message exchange. Event reports are generated as each limit is reached (one zone transition each), and when the counter is reset following calibration (one, two, or three zone transitions referenced in one report). Figure A7.3 illustrates this example. Note that, disabling the report upon counter reset (downward transitions) is not possible.

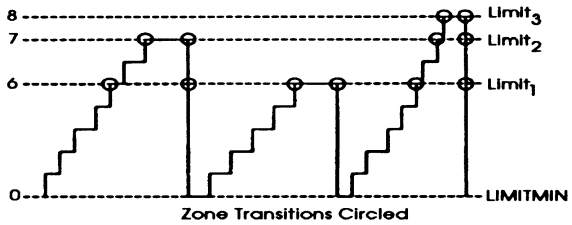



**Figure A7.3**
**Calibration Counter Example**

~~A7~~R1-7.2.4 *Example 4 — Derived Variables*

~~A7~~R1-7.2.4.1 The flagship of the ACME line is the new HotDog Furnace. This is a vertical furnace which exposes wafers to a variety of temperatures during processing. The temperature profile during the run is critical to the process and is typically contains a number of plateau's at different levels during the run. The owner wishes to monitor the temperature and be alerted whenever the actual temperature profile differs from the ideal by >0.5°. The derived variable was created to provide a steady target range during the run, no matter what the desired temperature range happened to be. Deviation from 'ideal temperature profile' was chosen as the new variable to be monitored. The equipment already had access to the profile for the run, which described the desired temperature at a given time into the process. The manufacturer added a calculation each time the actual temperature was sampled, subtracting the ideal temperature from the actual. They provided as status variables the actual temperature, the ideal temperature, and the new 'deviation from profile' variable. One limit was activated and set to 0.5° and a second set to −0.5° (each with a deadband width of 0.05). Thus, when the temperature deviation from setpoint exceeds ±0.5, an event is generated containing the current desired temperature and the actual temperature. For good measure, additional data was added, providing time since start of run to document the precise point in the process that the problem occurred.

~~A7~~R1-7.2.4.2 In order to achieve the desired behavior, the host defines four monitoring limits. Two of the limits establish the target zone. These are responsible for reporting transitions from normal to warning zones in either direction. The other two limits establish the transitions between the warning zones and the error zones. The difference between UPPERDB and LOWERDB for each limit is 0.5. This may also be expressed as limit ±0.25. Combining limits does not change the way the equipment treats limits monitoring, but rather builds a method of interpreting limits from the host's point of view.

## ~~A.8~~ R1-8 Process Parameter Modification for Process and Equipment Control

~~A8~~R1-8.1 *Introduction*

~~A8~~R1-8.1.1 In many equipment control applications there is a need for a GEM host to modify one or a small set of process parameters associated with a recipe. The number of parameters modified, frequency of modification (e.g., wafer-to-wafer, batch-to-batch, etc.), range of modification, etc., is largely a function of the equipment control application. Utilizing GEM, at least two methods are envisioned for modifying process parameters on a tool. With the first method, 'Equipment Constants' can be used to relate process parameters of the updated recipe. Equipment Constants can also be used in a mode where they relate suggested modifications to process parameters from the stored recipe; that is, the constants contain only the ± differential from a nominal value. The former mode is preferred because it better ensures data integrity between the controller and tool. With the second method the entire recipe could be downloaded, but this results in an enormous amount of communication overhead. Note that, in all cases, the Equipment Constants do not replace the process parameters inside a recipe, but are associated with (e.g., linked to) these

parameters to relate modifications. The remainder of this application note provides a description of how process parameter modification can be implemented using existing GEM capabilities. The method may be used in a GEM compliant system provided that the specific GEM capabilities described are supported.

~~A8~~R1-8.2 *Equipment Constants*

~~A8~~R1-8.2.1 Incremental process parameter modification for process and equipment control can be supported over a GEM interface by using the *Equipment Constants* GEM capability (see § 5.6). With this capability, each process parameter (or process parameter at a step) that can be modified (e.g., for purposes of process control) is associated with an equipment constant. Using the Equipment Constant GEM scenarios (see § 5.6.5) the host can (1), send process parameters or parameter modifications, (2), request current values of modifiable recipe parameters, (3), retrieve name lists of equipment constants associated with modifiable parameters, and (4), be informed by the equipment when an operator changes one of the modifiable process parameters.

~~A8~~R1-8.2.2 The equipment constants should represent the actual values of the process parameters with which they are associated. Depending on the equipment operation and control application, the equipment constant could represent the actual value of a process parameter at a recipe step, or over the entire recipe. The equipment constants could also be utilized to represent the differentials of process parameters from nominal values. However it is important to note that, when using differential values to relate process parameter modifications, any loss of synchronization between equipment and host could result in an incorrect assessment of the value of a process setpoint by the host. Note also that, upon system startup, and whenever the appropriate process parameters are modified, the equipment constants should also be modified as necessary to always reflect the (absolute or relative) values of the associated process parameters.

~~A8~~R1-8.2.3 In order to maintain synchronization between equipment and host, it is recommended that equipment constants associated with recipe parameters be applied to only override the currently selected and active recipe. A selected recipe is considered to be active whenever the equipment is in the ‘PROCESSING’ state and the recipe is the currently selected recipe (process program). If multiple recipes are utilized during one process event; for example, cluster tool scenario, it is recommended that separate equipment constants be utilized for each recipe/process parameter pair.

~~A8~~R1-8.2.4 Note that, timing and traceability issues associated with utilizing the equipment constants capability (for process control) are application specific and beyond the scope of this application note. Equipment that provides for recipe parameter overrides though setting of Equipment Contstants should also provide additional Equipment Constants for the set that includes the name of the associated process program and a Boolean variable to enable and disable the override feature. In addition, the supplier should document for each parameter: (1), the associated Equipment Constant, and (2), any restrictions on the state (active or not) or the recipe in which the parameter may be modified. Also, since override of the process setpoints may be provided by a Host controller application element, it is recommended that the equipment provide an event report whenever the associated recipe has been modified.

~~A8~~R1-8.3 *Example*

~~A8~~R1-8.3.1 In the example of Figure A.8.1, a Chemical-Mechanical Planarizer (CMP) single wafer ‘polishing’ system includes a GEM compliant planarizer (equipment), a thickness metrology unit and a (host) controller. The tool polishes a wafer to a target thickness. The post-process thickness is measured by the metrology unit and reported to the host controller. The controller utilizes a feedback control algorithm to determine the appropriate polish ‘time’ recipe parameter for the next wafer. This time should be reported to the CMP equipment utilizing the equipment’s GEM interface so that the information can be utilized for the next wafer processing event.

~~A8~~R1-8.3.2 The mechanism described in this application note could be utilized to implement process control as follows. A settable equipment constant is associated or ‘linked’ with the ‘time’ parameter on the equipment. Equipment system documentation indicates the linkage and the conditions under which the linkage is valid. When the controller determines an appropriate ‘time’ parameter value for the next wafer to be polished, it sets the equipment constant to this value (see § 5.6). The equipment is configured to accept this equipment constant change and, if the equipment is in (or possibly when the equipment reaches) the appropriate state, the recipe parameter is modified.

LETTER BALLOT

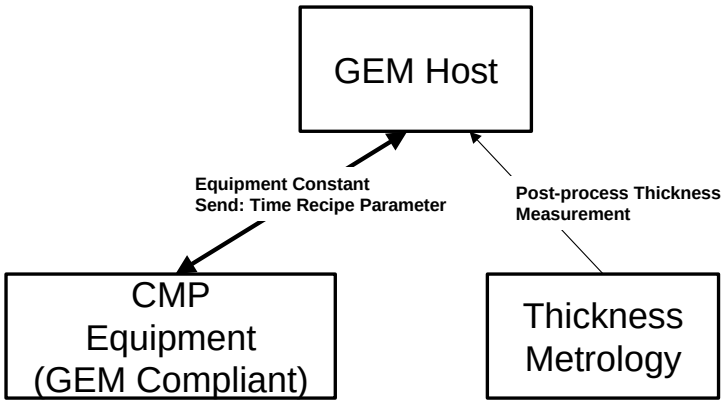



**Figure ~~A.8.1~~ R1-8.1**
**CMP Single Wafer 'Polishing' System with Host Recipe Parameter Modification Capability**

**NOTICE:** SEMI makes no warranties or representations as to the suitability of the Standards and Safety Guidelines set forth herein for any particular application. The determination of the suitability of the Standard or Safety Guideline is solely the responsibility of the user. Users are cautioned to refer to manufacturer's instructions, product labels, product data sheets, and other relevant literature, respecting any materials or equipment mentioned herein. Standards and Safety Guidelines are subject to change without notice.

By publication of this Standard or Safety Guideline, SEMI takes no position respecting the validity of any patent rights or copyrights asserted in connection with any items mentioned in this Standard or Safety Guideline. Users of this Standard or Safety Guideline are expressly advised that determination of any such patent rights or copyrights and the risk of infringement of such rights are entirely their own responsibility.

**<End of Ballot>**